

Cloudera Flow Management Upgrade and Migration

Date published: 2019-06-26

Date modified: 2023-11-02



Legal Notice

© Cloudera Inc. 2024. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Upgrade and migration paths.....	5
Upgrading from CFM 2.x to CFM 2.1.5 on CDP.....	8
Before you upgrade.....	8
Upgrading to CFM 2.1.5 from CFM 2.x.....	8
Upgrading from CFM 1.1.0 on CDH to CFM 2.1.5 on CDP.....	9
Before you upgrade.....	9
Turning off TLS regeneration.....	10
Backing up NiFi keystore and truststore settings.....	11
Backing up NiFi Registry keystore and truststore settings.....	12
Upgrading to CFM 2.1.5.....	13
Restoring NiFi keystore and truststore settings.....	14
Restoring your NiFi Registry keystore and truststore settings.....	15
Turning off identity mapping.....	16
Additional post-upgrade steps for some upgrade scenarios.....	17
Enabling Auto-TLS for CFM.....	17
Creating a Ranger user for the Initial Admin Identity.....	18
Integrating with Atlas manually.....	19
Integrating with Atlas when Auto-TLS is enabled.....	20
Starting your NiFi and NiFi Registry services.....	20
Migration from HDF 3.5.x or CFM 1.1.0 to CFM 2.x on CDP.....	20
Before you begin.....	21
Preserving source cluster files and directories.....	21
Preserving custom processors/NARs.....	21
NiFi files to preserve.....	22
NiFi Registry files to preserve.....	22
Installing CFM 2.1.5.....	23
Adding and configuring the NiFi service.....	23
Adding and configuring the NiFi Registry service.....	25
Verifying CFM 2.1.5.....	26
Clearing activity and shutting down source services.....	27
Migrating the NiFi data directories.....	28
Migrating the NiFi flow.xml.gz file.....	29
Updating encryption algorithm for sensitive properties.....	29
Removing unnecessary reporting tasks.....	29
Updating the Registry Client.....	31
Updating references to cluster nodes.....	31
Updating a flow with sensitive properties.....	32
Migrating authorization policies.....	33
Migrating NiFi Ranger-based policies.....	33
Migrating NiFi Registry Ranger-based policies.....	34
Migrating NiFi file-based policies.....	34
Migrating NiFi Registry file-based policies.....	36
Migrating NiFi state and custom components.....	37

Migrating NiFi Registry data storage.....	38
Migrating the metadata database.....	38
Migrating flow storage.....	39
Migrating bundle storage configurations.....	42
Reviewing NiFi components.....	44
Migrating file-based authorization to Ranger.....	44
Migrating NiFi File-Based authorization to Ranger.....	44
Migrating NiFi Registry file-based authorization to Ranger.....	45

In-place upgrade from HDF 3.5.x to CFM 2.x on CDP.....45

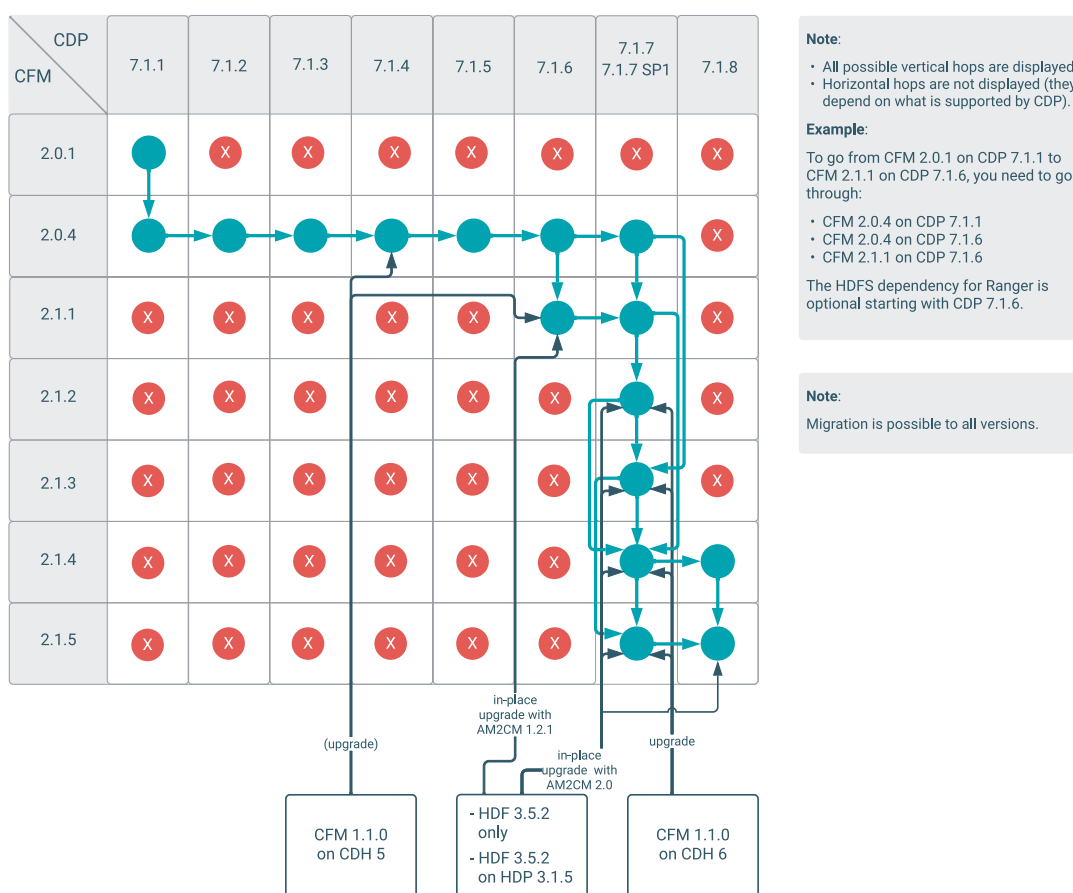
Before you upgrade.....	45
Setting Ambari environment variables.....	45
Installing Solr service.....	47
Checking cluster services.....	47
Checking service accounts.....	48
Collecting data for upgrade.....	48
Downloading and extending Ambari blueprint.....	49
Downloading Ambari blueprint.....	49
Extending the JSON file.....	50
Installing and setting up Cloudera Manager.....	50
Checking preinstallation setup.....	50
Configuring Cloudera Manager repository.....	51
Installing Cloudera Manager server and agents.....	52
Configuring database for Cloudera Manager.....	52
Starting Cloudera Manager server and adding license.....	53
Configuring Cloudera agents and hosts.....	54
Adding and configuring Cloudera Management Services.....	56
Upgrading HDF to CFM on CDP Private Cloud Base.....	59
Generating Cloudera Manager Deployment template.....	59
Configuring parcels.....	61
Deploying Cloudera Manager.....	62
Adding CFM parcel in Cloudera Manager.....	63
Activating parcel.....	64
Troubleshooting HDF upgrade.....	64
Post-upgrade steps on CDP.....	65
Enabling security.....	66
Setting core configuration service.....	66
Starting Zookeeper service.....	68
Configuring NiFi Registry settings.....	68
Verifying Ranger configurations.....	69
Configuring Ranger settings.....	69
Configuring Solr settings.....	71
Initializing Solr.....	72
Configuring NiFi settings.....	73
Configuring YARN settings.....	75
Migrating Kafka Ranger policies.....	76

Upgrade and migration paths

To use the latest enhancements and capabilities of NiFi, Cloudera recommends deploying it on the Cloudera Data Platform (CDP). You can seamlessly integrate NiFi with CDP by installing the Cloudera Flow Management (CFM) parcel on your CDP infrastructure.

This guide provides information on upgrading or migrating from various NiFi deployment scenarios, including HDF, CFM on CDH, or earlier versions of CFM on CDP Private Cloud Base. Review the upgrade and migration details to ensure that you understand the specific requirements relevant to your chosen upgrade or migration scenario, aligning them with your unique use case.

Upgrade and migration paths overview



From CFM 2.x on CDP Private Cloud Base

If you are using CFM 2.x on CDP Private Cloud Base (CDP PvC Base) version 7.1.1 – 7.1.8, and you intend to upgrade to a higher CFM version, there are specific upgrade paths to follow. Check the [Upgrade paths diagram](#) and refer to the step-by-step procedure outlined in the following table:

Table 1: Upgrade procedure from CFM 2.x

	Upgrade
Step 1	Upgrade CFM to 2.0.4 if you are using CFM 2.0.1 on CDP PvC Base 7.1.1
Step 2	Upgrade CDP Private Cloud Base to 7.1.7, 7.1.7 SP1, or 7.1.7 SP2
Step 3	Upgrade CFM to a higher version
Step 4	Upgrade CDP Private Cloud Base to 7.1.8



Note: CFM 2.1.1 is also supported on CDP Private Cloud Base 7.1.6.

From CFM 1.x on CDH

If you are using CFM 1.x on CDH 5 or CDH 6, or as standalone CFM, and you want to upgrade to a higher CFM version, there are specific upgrade paths to follow. Check the [Upgrade paths diagram](#) and refer to the step-by-step procedure outlined in the following table:

Table 2: Upgrade/migration options from CFM 1.x

	Migration		Upgrade from CFM 1.x on CDH 6		Upgrade from CFM 1.x on CDH 5		Upgrade from standalone CFM 1.x	
Step 1	Upgrade CFM 1.0.0 to 1.1.0	Upgrade CFM 1.0.1 to 1.1.0	Upgrade CFM 1.0.0 to 1.1.0	Upgrade CFM 1.0.1 to 1.1.0	Upgrade CFM 1.0.0 to 1.1.0	Upgrade CFM 1.0.1 to 1.1.0	Upgrade CFM 1.0.0 to 1.1.0	Upgrade CFM 1.0.1 to 1.1.0
Step 2	Install CFM 2.x on CDP Private Cloud Base 7.1.1 – 7.1.8		Upgrade CDH 6 to CDP Private Cloud Base 7.1.7, 7.1.7 SP1, or 7.1.7 SP2		Upgrade CDH 5.0-5.12 to CDH/CM 5.13 or higher For instructions, see the Cloudera Enterprise Upgrade Guide.		Upgrade CFM 1.1.0 to CFM 2.x	
Step 3	Migrate data from CFM 1.x to CFM 2.x		Upgrade from CFM 1.1.0 to CFM 2.x		Upgrade CDH 5.13 or higher to CDP Private Cloud Base 7.1.1 or 7.1.7			
Step 4					Upgrade CFM 1.1.0 to CFM 2.x			



Note: Cloudera recommends migration over in-place upgrade.

For a better understanding of the differences between upgrade and migration, see [CFM upgrade and migration options](#).

From HDF

If you are using HDF and you want to upgrade to CFM, there are specific upgrade paths to follow. Check the [Upgrade paths diagram](#) and refer to the step-by-step procedure outlined in the following table:

Table 3: Upgrade/migration options from HDF

	Migration	In-place upgrade
Step 1	Upgrade HDF to 3.5.x See also the HDF 3.5.2 Release Notes .	Upgrade HDF to 3.5.2 See also the HDF 3.5.2 Release Notes .
Step 2	Install CFM 2.x on CDP Private Cloud Base 7.1.1 – 7.1.8	In-place upgrade HDF 3.5.2 to CFM 2.1.1 on CDP PvC Base 7.1.6 or CFM 2.1.2 – 2.1.4 on CDP PvC Base 7.1.7.
Step 3	Migrate data from HDF to CFM	



Note: Cloudera recommends migration over in-place upgrade.

For a better understanding of the differences between upgrade and migration, see [CFM upgrade and migration options](#).

CFM upgrade and migration options

There are three methods available for transitioning from an older version to CFM 2.x:

Upgrade

Upgrade refers to a full upgrade of CFM on CDP Private Cloud Base without the necessity for migrating the data between clusters.

Possible upgrade paths include:

- CFM 1.1.0 CFM 2.x
For instructions, see [Upgrading from CFM 1.x](#).
- CFM 2.x CFM 2.x
For instructions, see [Upgrading from CFM 2.x](#).



Important: For precise version-to-version upgrade paths, see the [Upgrade paths diagram](#).

Migration

Data migration involves the transfer of existing HDF or CFM 1.x cluster workloads to a fresh installation of CDP Private Cloud Base. You can migrate your NiFi dataflows and NiFi Registry versioned flows from an HDF 3.5.x or CFM 1.1.0 (standalone, or on CDH 5 / CDH 6) cluster to a CFM 2.x cluster on CDP PvC Base. This method requires that you have both your HDF or CFM 1.1.0 cluster and a CFM 2.x cluster running at the same time.



Note: Cloudera recommends migration over in-place upgrade.

For detailed instructions on performing migration, see [Migration from HDF 3.5.x or CFM 1.x](#).

Possible migration paths include:

- HDF 3.5.x CFM 2.x
- CFM 1.1.0 CFM 2.x



Important: For precise version-to-version upgrade paths, see the [Upgrade paths diagram](#).

In-place upgrade

This option allows you to install Cloudera Manager when using HDF and then enable Cloudera Manager to take over the management of your services. You can transform your existing Ambari blueprint into a Cloudera Manager Deployment template using the AM2CM tool.



Note: Cloudera recommends migration over in-place upgrade.

If you have a version of HDF lower than 3.5.2.0, first upgrade to HDF 3.5.2.0 and then upgrade HDF to CFM 2.1.1 – 2.1.4 on CDP PvC Base 7.1.6 or 7.1.7. For detailed instructions on performing an in-place upgrade, see [In-place upgrade from HDF 3.5.x](#).

Possible in-place upgrade paths include:

- HDF 3.5.2.0 CFM 2.1.1 on CDP 7.1.6
- HDF 3.5.2.0 CFM 2.1.1-2.1.5 on CDP 7.1.7



Important: For precise version-to-version upgrade paths, see the [Upgrade paths diagram](#).

Related Information

[Use case examples on how to migrate flows from HDF to CFM with no downtime](#)

[No Data Loss and No Service Interruption -- HDF to CFM Rolling Migration](#)

[Supported CDP Private Cloud Base in-place upgrade paths](#)

Upgrading from CFM 2.x to CFM 2.1.5 on CDP

This guide outlines the prerequisites and steps of the upgrade process from a lower version of Cloudera Flow Management (CFM) 2.x to CFM 2.1.5.

You can upgrade to CFM 2.1.5 from CFM 2.1.4 and CFM 2.1.3. If you want to upgrade from other CFM 2.x versions, check the available upgrade paths in [Upgrade and migration paths](#).

Before you upgrade

Before starting your CFM upgrade, it is essential that you review the migration and upgrade options, upgrade paths, and the CDP Private Cloud Base cluster requirements to ensure that you understand and fulfill the prerequisites.

Upgrade paths

You can upgrade to CFM 2.1.5 from the following previous CFM versions:

- CFM 2.1.4
- CFM 2.1.3

To upgrade to CFM 2.1.5 from other CFM 2.x versions, check the available upgrade paths in [Upgrade and migration paths](#).

Supported CDP Private Cloud Base cluster versions

CFM 2.1.5 runs on the following CDP Private Cloud Base cluster versions:

- 7.1.7
- 7.1.7 SP1
- 7.1.7 SP2
- 7.1.8



Important:

You cannot run CFM 2.1.5 on CDH 5.x or 6.x clusters. To upgrade your underlying cluster, see the CDP upgrade documentation appropriate for your use case.

- [CDP Upgrade and Migrations Paths](#)
- [Upgrading CDH 6 to CDP Private Cloud Base](#)
- [Upgrading CDP Private Cloud Base to a higher version](#)

Upgrading to CFM 2.1.5 from CFM 2.x

To upgrade to CFM 2.1.5 from a lower version of CFM on CDP Private Cloud Base, you must stop the CFM services, update the CSD files, restart the SCM Server, activate the new CFM parcel, and then restart your CFM services.

Before you begin

- You have upgraded Cloudera Manager and CDP Private Cloud Base if needed.



Important:

CFM 2.1.5 requires at least CDP Private Cloud Base 7.1.7.

- You have reviewed the [Upgrade and migration paths](#) to make sure you are performing the right upgrade for your use case.
- You have reviewed the [Before you upgrade](#) information to make sure you meet the prerequisites.

Procedure

1. Stop the NiFi and NiFi Registry services, in this order.
2. Delete the old CSD files.
3. Download the new CSD files.

The default CSD location is `/opt/cloudera/csd`, but you can configure that location from Cloudera Manager Administration | Settings | Local Descriptor Repository Path.

Ensure that you maintain the appropriate file ownership and access attributes and SELinux permissions if needed.

4. Download the new CFM parcel and .sha checksums file appropriate for your operating system.

The default parcel location is `/opt/cloudera/parcel-repo/`.

Ensure that you maintain the appropriate file ownership and access attributes and SELinux permissions if needed.

5. Restart the `cloudera-scm-server` service:

```
service cloudera-scm-server restart
```

6. In Cloudera Manager, from the **Parcels** page, distribute and activate the CFM 2.1.5 parcel.

After clicking Activate, the Activate CFM <version> on <cluster-name> pop-up displays. Click Activate Only, and then OK.

7. Restart Cloudera Management Service.
8. Restart NiFi and NiFi Registry and deploy the client configurations.
9. Optionally, to clean up, remove the previous parcels from the host.
 - a) From the Parcel menu, go to the older CFM parcel.
 - b) Select Remove From Host.
 - c) Select Delete.
10. From the Ranger UI, NiFi service controller policy:
 - a) Add the `nifi` user to the policy with READ permissions.
 - b) Verify the `nifi` group is set in the policy.

Upgrading from CFM 1.1.0 on CDH to CFM 2.1.5 on CDP

This section describes the steps of the upgrade process from CFM 1.1.0 on CDH to CFM 2.1.5 on CDP Private Cloud Base.

Before you upgrade

Before starting your CFM upgrade, it is crucial that you review the migration and upgrade options, upgrade paths, and your CDP Private Cloud Base cluster requirements to ensure that you understand and fulfill the prerequisites.

Upgrade paths

If you have CFM 1.0.x, first you need to upgrade to CFM 1.1.0.

If you have CFM 1.1.0 running on CDH 6.x, you can upgrade to CFM 2.1.5. For other upgrade scenarios, see the [Upgrade and migration paths](#).

Supported CDP Private Cloud Base cluster versions

CFM 2.1.5 runs on the following CDP Private Cloud Base cluster versions:

- 7.1.7
- 7.1.7 SP1
- 7.1.7 SP2
- 7.1.8



Important:

You cannot run CFM 2.1.5 on CDH 5.x or CDH 6.x clusters. To upgrade your underlying cluster, see the CDP upgrade documentation appropriate for your use case.

- [CDP Upgrade and Migrations Paths](#)
- [Upgrading CDH 6 to CDP Private Cloud Base](#)
- [Upgrading CDP Private Cloud Base to a higher version](#)

Turning off TLS regeneration

The NiFi CA is not installed as part of CFM 2.1.5. You must turn off NiFi CFM upgrade edits CA Force Regenerate before proceeding with your upgrade.

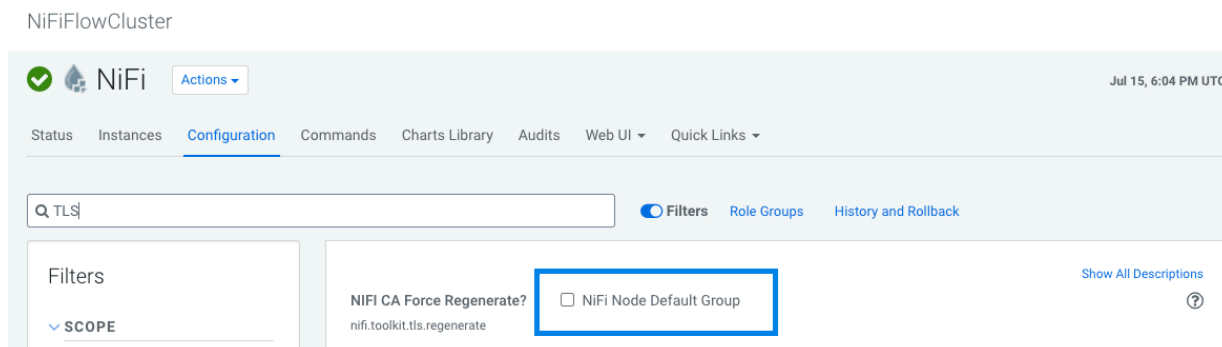
Before you begin

You have reviewed the following *Before you upgrade* information and are sure that you are performing the right upgrade for your use case.

- CFM migration and upgrade options
- Upgrade paths
- Supported CDP Private Cloud Base cluster versions

Procedure

1. From Cloudera Manager, click the Clusters tab in the left-hand navigation
2. Click NiFi in the list of services to display the NiFi service page.
3. Select the Configuration tab.
4. Deselect the TLS regeneration check-box.



5. Repeat these steps for NiFi Registry.

What to do next

Once you have turned off TLS regeneration, back up your keystore and truststore values for NiFi and NiFi Registry, and then proceed with the upgrade to CFM 2.1.5.

Once you have completed the upgrade, Cloudera recommends that you use Auto-TLS for your CDP Private Cloud Base cluster.

Related Information

[Before you upgrade](#)

[Backing up NiFi keystore and truststore settings](#)

Backing up NiFi keystore and truststore settings

If your CFM installation from which you are upgrading is TLS enabled, use the Encrypt Config tools to back up your NiFi keystore and truststore settings. You will set these values in Cloudera Manager once you complete the upgrade.

Before you begin

- You have turned off TLS regeneration.
- If JAVA_HOME is not set, you should set it before proceeding. The default path is /usr/java/default.

Procedure

1. Locate the `encrypt-config.sh` script from the NiFi Toolkit.

The default location is /opt/cloudera/parcels. You can find your location by running:

```
find /opt/cloudera/parcels -name 'encrypt-config.sh'
```



Note:

If you have installed more than one CFM parcel, you may have more than one script. In this case, ensure that you have the script from CFM 1.1.0.

2. Find the latest NiFi process directory:

```
find /var/run/cloudera-scm-agent/process/ -name nifi.properties | grep "NIFI_NODE"
```

3. Run `encrypt-config.sh`:

```
<path_to_encrypt-config.sh>
-c
-b <path_to_nifi_proc_dir>/bootstrap.conf
-n <path_to_nifi_proc_dir>/nifi.properties
```

For example:

```
/opt/cloudera/parcels/CFM-1.1.0.0/encrypt-config.sh
-c
-b /run/cloudera-scm-agent/182-NIFI_NODE.../bootstrap.conf
-n /run/cloudera-scm-agent/182-NIFI_NODE.../nifi.properties
```

4. Back up the `encrypt-config.sh` output.

Results

The `encrypt-config.sh` output will be similar to:

```
keystore=/var/lib/nifi/cert/keystore.jks
keystorePasswd=/TLVwnnFESyIwn2YrBGiVWrANNhiSk
keyPasswd=/TLVwnnFESyIwn2YrBGiVWrANNhiSk
truststore=/var/lib/nifi/cert/truststore.jks
truststorePasswd=4wIWsnHpkVa5MR8P353s3ruMDGj1UL
```

What to do next

Once you have completed this step for NiFi, do the same for NiFi Registry.

Related Information

[Turning off TLS regeneration](#)

[Backing up NiFi Registry keystore and truststore settings](#)

Backing up NiFi Registry keystore and truststore settings

If your CFM installation from which you are upgrading is TLS enabled, use the Encrypt Config tools to backup your NiFi Registry keystore and truststore settings. You will set these values in Cloudera Manager once you complete the upgrade.

Before you begin

If `JAVA_HOME` is not set, you should set it before proceeding. The default path is `/usr/java/default`.

Procedure

1. Locate the `encrypt-config.sh` script from the NiFi Toolkit.

The default location is `/opt/cloudera/parcels`. You can find your location by running:

```
find /opt/cloudera/parcels -name 'encrypt-config.sh'
```



Note:

If you have installed more than one CFM parcel, you may have more than one script. In this case, ensure that you have the script from CFM 1.1.0.

2. Find the latest NiFi Registry process directory:

```
find /var/run/cloudera-scm-agent/process/ -name nifi-registry.properties
| grep "NIFI_REGISTRY_SERVER"
```

3. Run `encrypt-config.sh`:

```
${ENCRYPT_CONFIG_PATH}
--nifiRegistry
--decrypt
-r ${NIFIREG_PROC_DIR}/nifi-registry.properties
-b ${NIFIREG_PROC_DIR}/bootstrap.conf
```

4. Back up the `encrypt-config.sh` output.

Results

The `encrypt-config.sh` output will be similar to:

```
nifi.registry.security.keystore=/var/lib/nifiregistry/cert/keystore.jks
nifi.registry.security.keystorePasswd=5BNrrLRmcrgi+qq1BNpEpoIzyOALo
nifi.registry.security.truststore=/var/lib/nifiregistry/cert/truststore.jks
nifi.registry.security.truststorePasswd=qKdbQ9Q0a0uX/XApHhLjR4d2zxRHQ3
```



Note:

`nifi.registry.security.keystorePasswd` is the same as `keyPassword`.

What to do next

Once you have completed this step for NiFi Registry, you may proceed with the upgrade to CFM 2.1.5.

Related Information

[Backing up NiFi keystore and truststore settings](#)

[Upgrading to CFM 2.1.5](#)

Upgrading to CFM 2.1.5

To upgrade from CFM 1.0.x to CFM 2.1.5, you need to upgrade to CFM 1.1.0 first. Then you must stop the CFM services, update the CSD files, restart the SCM Server, and activate the new CFM parcel.

About this task

If you are upgrading from CFM 1.0.0 or CFM 1.0.1, first you need to upgrade to CFM 1.1.0.

If you are upgrading from CFM 1.1.0 on CDH 6.x, perform the following steps.

For other upgrade scenarios, see [Upgrade and migration paths](#).

Before you begin

Before you begin the CFM upgrade, ensure that you have completed the steps to:

- Turn off TLS regeneration.
- Back up your keystore and truststore settings for NiFi and NiFi Registry.

Procedure

1. Stop NiFi, NiFi Registry, and NiFi CA Service in this order.
2. Delete the old CSD files.
3. Download the new CSD files.

The default CSD location is `/opt/cloudera/csd`, but you can configure that location from Cloudera Manager Administration | Settings | Local Descriptor Repository Path.

Ensure that you maintain the appropriate file ownership and access attributes and SELinux permissions if needed.

4. Download the new CFM parcel and .sha checksum file appropriate for your operating system.

The default parcel location is `/opt/cloudera/parcel-repo/`.

Ensure that you maintain the appropriate file ownership and access attributes and SELinux permissions if needed.

5. Restart the `cloudera-scm-server` service:

```
service cloudera-scm-server restart
```

6. In Cloudera Manager, from the **Parcels** page, distribute and activate the CFM parcel to which you want to upgrade.

After clicking Activate, the Activate CFM <version-number> on <cluster-name> pop-up displays. Click Activate Only, and then OK.

7. Optionally, to clean up, remove the older parcels from the host.

- a) From the Parcel menu, go to the older CFM parcel.
- b) Select Remove From Host.
- c) Select Delete.

**Important:**

Do not start the NiFi and NiFi Registry services yet. Complete the post-upgrade steps before you start any services.

What to do next

Once you have completed the upgrade, perform the following additional tasks:

- Set the keystore and truststore settings with the values from your backup.
- Turn off identity mapping for both NiFi and NiFi Registry.
- Start NiFi and NiFi Registry.

**Note:**

Do not start the NiFi CA service, as this is no longer supported as part of CFM 2.1.5.

- Optionally, you can remove the NiFi CA service.

Related Information

[Upgrading CFM 1.0.x](#)

[Backing up NiFi keystore and truststore settings](#)

[Backing up NiFi Registry keystore and truststore settings](#)

Restoring NiFi keystore and truststore settings

Learn how to restore your NiFi keystore and truststore settings from the backup you made prior to upgrade.

Before you begin

- You have completed your upgrade to CFM 2.1.5.
- You have the NiFi keystore and truststore settings that you backed up before beginning your upgrade.

Procedure

1. From Cloudera Manager, click the Clusters tab in the left-hand navigation.
2. Click NiFi in the list of services to display the NiFi service page.
3. Select the Configuration tab.




- Use the search bar to find the Keystore configuration options and update the following three with the values from your backup.

**Note:**

The `nifi.security.keystorePasswd` value should be the same as the `keyPassword` value.

NiFi Node TLS/SSL Server JKS Keystore File Location <small>nifi.security.keystore</small>	NiFi Node Default Group 	
	<input type="text" value="{{CM_AUTO_TLS}}"/>	
NiFi Node TLS/SSL Server JKS Keystore File Password <small>nifi.security.keystorePasswd</small>	NiFi Node Default Group 	
	<input type="password" value="....."/>	
NiFi Node TLS/SSL Server JKS Keystore Key Password <small>nifi.security.keyPasswd</small>	NiFi Node Default Group 	
	<input type="password" value="....."/>	

- Use the search bar to find the Truststore configuration options and update the following two with the values from your backup.

NiFi Node TLS/SSL Client Trust Store File <small>nifi.security.truststore</small>	NiFi Node Default Group 	
	<input type="text" value="{{CM_AUTO_TLS}}"/>	
NiFi Node TLS/SSL Client Trust Store Password <small>nifi.security.truststorePasswd</small>	NiFi Node Default Group 	
	<input type="password" value="....."/>	

What to do next

Once you have completed this step for the NiFi service, do the same for NiFi Registry.

Related Information

[Upgrading to CFM 2.1.5](#)

[Restoring your NiFi Registry keystore and truststore settings](#)

Restoring your NiFi Registry keystore and truststore settings

Learn how to restore your NiFi Registry keystore and truststore settings from the backup you made prior to upgrade.

Before you begin

- You have completed your upgrade to CFM 2.1.5.
- You have the NiFi Registry keystore and truststore settings that you backed up before beginning your upgrade.
- You have restored NiFi keystore and truststore settings.

Procedure

- From Cloudera Manager, click the Clusters tab in the left-hand navigation
- Click NiFi Registry in the list of services to display the NiFi Registry service page.
- Select the Configuration tab.

- Use the search bar to find the Keystore configuration options and update the following three with the values from your backup.

**Note:**

The `nifi.registry.security.keystorePasswd` value should be the same as the `keyPassword` value.

NiFi Registry TLS/SSL Server	NiFi Registry Default Group	?
JKS Keystore File Location	<input type="text" value="{{CM_AUTO_TLS}}"/>	
<small>nifi.registry.security.keystore</small>		
NiFi Registry TLS/SSL Server	NiFi Registry Default Group	?
JKS Keystore File Password	<input type="password" value="....."/>	
<small>nifi.registry.security.keystorePasswd</small>		
NiFi Registry TLS/SSL Server	NiFi Registry Default Group	?
JKS Keystore Key Password	<input type="password" value="....."/>	
<small>nifi.registry.security.keyPasswd</small>		

- Use the search bar to find the Truststore configuration options and update the following two with the values from your backup.

NiFi Registry TLS/SSL Client	NiFi Registry Default Group	?
Trust Store File	<input type="text" value="{{CM_AUTO_TLS}}"/>	
<small>nifi.registry.security.truststore</small>		
NiFi Registry TLS/SSL Client	NiFi Registry Default Group	?
Trust Store Password	<input type="password" value="....."/>	
<small>nifi.registry.security.truststorePasswd</small>		

What to do next

Once you have restored NiFi and NiFi Registry keystore and truststore settings, turn off identity mapping.

Related Information

[Restoring NiFi keystore and truststore settings](#)

[Turning off identity mapping](#)

Turning off identity mapping

Describes the steps to turn off identity mapping.

About this task

If you enable auto-TLS, you must ensure that identity mapping is turned off.

Before you begin

You have restored NiFi and NiFi Registry keystore and truststore settings.

Procedure

- From Cloudera Manager, click the Clusters tab in the left-hand navigation.
- Click NiFi in the list of services to display the NiFi service page.

3. Select the Configuration tab.
4. Use the search bar to find the Identity Mapping configuration options and remove the values for the following parameters:
 - Identity Mapping - DN Pattern (`nifi.security.identity.mapping.pattern.dn`)
 - Identity Mapping - DN Value (`nifi.security.identity.mapping.value.dn`)
5. Repeat these steps for NiFi Registry. The NiFi Registry Identity Mapping configuration options are:
 - Identity Mapping - DN Pattern (`nifi.registry.security.identity.mapping.pattern.dn`)
 - Identity Mapping - DN Value (`nifi.registry.security.identity.mapping.value.dn`)

Example

The screenshot shows the Cloudera Manager interface for NiFi configuration. The search bar contains 'identity map'. The results are displayed in a table with two columns: 'Identity Mapping - DN Pattern' and 'Identity Mapping - DN Value'. The 'Identity Mapping - DN Pattern' row shows the parameter `nifi.security.identity.mapping.pattern.dn` with a value of `^CN=(.*?),.+$.`. The 'Identity Mapping - DN Value' row shows the parameter `nifi.security.identity.mapping.value.dn` with a value of `$1`. The interface also includes a left sidebar with filters, a top navigation bar with tabs like Status, Instances, Configuration, Commands, Charts Library, Audits, Web UI, and Quick Links, and a top right corner showing the date and time.

What to do next

Once you have turned off identity mapping, review the additional post-upgrade steps for any additional requirements that pertain to your CFM 2.1.5 deployment scenario.

Related Information

[Restoring your NiFi Registry keystore and truststore settings](#)

[Additional post-upgrade steps for some upgrade scenarios](#)

Additional post-upgrade steps for some upgrade scenarios

Depending on the type of CFM 1.1.0 installation you are upgrading, there are some additional post-upgrade steps you must take. You should review the following information for anything pertaining to your upgrade scenario.

Enabling Auto-TLS for CFM

Provides steps to enable Auto-TLS for CFM.

About this task

You should perform these steps if you are upgrading from a CFM 1.1.0 installation where:

- TLS is enabled for CFM 1.1.0; AND
- Auto-TLS is enabled on the CDH cluster.

Before you begin

- You have turned off identity mapping.

Procedure

1. Launch the API Explorer from the Cloudera Manager Support menu at the bottom of the left navigation pane.

2. Run the `configureAutoTlsServices` API call.

The screenshot shows the API interface for the `POST /clusters/{clusterName}/commands/configureAutoTlsServices` endpoint. The description states: "Configures all services in a cluster to use Auto-TLS." Under "Implementation Notes", it says "Configures all services in a cluster to use Auto-TLS." The "Response Class (Status 201)" is "Success". A "Model Schema" is displayed as a JSON object with fields: `id` (12345), `name` (...), `startTime` (...), `endTime` (...), `active` (true), `success` (true), `resultMessage` (...), `resultDataUrl` (...), and `clusterRef` (a nested object). Below the schema, the "Response Content Type" is set to `application/json`. There are sections for "Headers" and "Parameters". The "Parameters" section includes a table with one parameter: `clusterName` (required), described as "The name of the cluster.", with a "path" parameter type and a "string" data type. A "Try it out!" button is at the bottom.

3. Edit the `users.xml`, to remove the users associated with the NiFi nodes.

Repeat these steps for NiFi Registry.

4. Edit the `authorizations.xml` file.

In the `/proxy` policy, remove the users corresponding to the NiFi nodes and replace them with:

```
<group identifier="nifi"/>
```

Repeat these steps for NiFi Registry/

5. Review the other policies related to the NiFi nodes, to similarly edit any other references to the NiFi nodes.

What to do next

Once you have enabled auto-TLS, create a Ranger user for the Initial Admin Identity.

Related Information

[Turning off identity mapping](#)

[Creating a Ranger user for the Initial Admin Identity](#)

Creating a Ranger user for the Initial Admin Identity

In some upgrade scenarios, you must manually create a Ranger user for the Initial Admin Identity and add it to the `nifi` group.

About this task

You should perform these steps if you are upgrading from a CFM 1.1.0 installation where:

- CFM has Kerberos and TLS enabled; AND
- The CDP Private Cloud Base cluster does not have Auto-TLS enabled; AND
- You want Ranger as part of your CFM 2.1.5 on CDP Private Cloud Base 7.1.x deployment.

Before you begin

Ranger is running on your CDP Private Cloud Base 7.1.x cluster.

Procedure

1. Create a Ranger user with the same user name as your NiFi Initial Admin Identity.
2. Assign this user to the Ranger group `nifi`.
3. Create a Ranger user with the same username as your NiFi Registry Initial Admin Identity.
4. Assign this user to the Ranger group `nifiregistry`.

Related Information

[Enabling Auto-TLS for CFM](#)

Integrating with Atlas manually

Provides steps to manually integrate with Atlas by creating the `ReportLineageToAtlas` reporting task.

About this task

If you are upgrading from a CFM 1.1.0 installation where:

- CFM does not have TLS enabled; AND
- The CDP Private Cloud Base cluster does not have Auto-TLS enabled; AND
- You do not want to enable Auto-TLS; AND
- You want Atlas as part of CFM 2.1.5 on your CDP Private Cloud Base 7.1.x deployment.

Procedure

1. Start NiFi.
 - a) From Cloudera Manager, click the Clusters tab in the left-hand navigation.
 - b) Click NiFi in the list of services to display the NiFi service page.
 - c) Click the Actions drop-down, and then click Start.
2. From the Global Menu located in NiFi's upper right corner, select Controller Services and click the Reporting Tasks tab.
3. Click the Add (+) icon to launch the Add Reporting Task dialog.
4. Select `ReportLineageToAtlas` and click Add.
5. Click the Edit icon to launch the Configure Reporting Task dialog. The following properties are required:
 - Atlas URLs – a comma-separated list of Atlas Server URLs. Once you have started reporting, you cannot modify an existing Reporting Task to add a new Atlas Server. When you need to add a new Atlas Server, you must create a new reporting task.
 - Atlas Authentication Method – Specifies how to authenticate the Reporting Task to the Atlas Server. Basic authentication is the default.
 - NiFi URL for Atlas – Specifies the NiFi cluster URL
 - Lineage Strategy – Specifies the level of granularity for your NiFi dataflow reporting to Atlas. Once you have started reporting, you should not switch between simple and complete lineage reporting strategies.
 - Provenance Record Start Position – Specifies where in the Provenance Events stream the Reporting Task should start.
 - Provenance Record Batch Size – Specifies how many records you want to send in a single batch
 - Create Atlas Configuration File – If enabled, the `atlas-application-properties` file and the Atlas Configuration Directory are automatically created when the Reporting Task starts.
 - Kafka Security Protocol – Specifies the protocol used to communicate with Kafka brokers to send Atlas hook notification messages. This value should match Kafka's `security.protocol` property value.

Integrating with Atlas when Auto-TLS is enabled

Provides manual steps to integrate with Atlas when Auto-TLS is enabled on your CDP Data Center cluster.

About this task

You must perform these steps if:

- You want CFM 2.1.5 to integrate with Atlas; AND
- The CDP Private Cloud Base 7.1.x cluster has Auto-TLS enabled

Procedure

1. Start NiFi.
 - a) From Cloudera Manager, click the Clusters tab in the left-hand navigation.
 - b) Click NiFi in the list of services to display the NiFi service page.
 - c) Click the Actions drop-down, and then click Start.
2. Select the Atlas Dependency d checkbox.
3. Restart NiFi.
4. Click Create required NiFi objects from the Actions drop-down.

Starting your NiFi and NiFi Registry services

After performing all the upgrade and possible post-upgrade steps, start your NiFi and NiFi Registry services.

About this task

The final upgrade step is to start your NiFi and NiFi Registry services if you have not already done so.

Before you begin

You have completed all applicable post-upgrade steps. If you have not completed these steps, your services may fail to start.

Procedure

1. From Cloudera Manager, click the Clusters tab in the left-hand navigation.
2. Click NiFi in the list of services to display the NiFi service page.
3. Click the Actions drop-down, and then click Start.
4. Repeat these steps for NiFi Registry.

Related Information

[Additional post-upgrade steps for some upgrade scenarios](#)

Migration from HDF 3.5.x or CFM 1.1.0 to CFM 2.x on CDP

This section describes the general steps required to move your NiFi dataflows and NiFi Registry versioned flows from an HDF 3.5.x or CFM 1.1.0 standalone cluster to a CFM 2.1.5 cluster on CDP Private Cloud Base.



Note: If you are using an HDF version lower than 3.5.x, you need to upgrade to HDF 3.5.x first and then proceed with the migration to CFM. For details, see [Ambari Managed HDF Upgrade](#).

Before you begin

Check the possible migration paths and requirements before you start your migration process.

Migration paths

You can migrate data from:

- HDF 3.5.x
 - Standalone (not managed by Ambari)
 - Ambari (with or without Ranger)
 - +HDP 3.1.5 and Ambari (with or without Ranger)
- CFM 1.1.0
 - Standalone (not managed by Cloudera Manager)

For details on data migration and other upgrade options, see [Upgrade and migration paths](#).

Additional requirements



Note: HDF and CFM are packaged with a different component list. For details, see [CFM component versions](#) and [HDF Component Support](#). Components that are part of HDF but not part of CFM will become part of the CDP package. For information on migrating workloads of HDF components that are not included in CFM into CDP, see [Migrating Workloads](#) in the *CDP Private Cloud Upgrade Guide*.

- You must migrate HDF 3.5.x or CFM 1.1.0 standalone to an equivalent target environment.

For example:

- From HDF 3.5.x without Ranger to CFM 2.1.5 without Ranger
- From HDF 3.5.x with Ranger to CFM 2.1.5 with Ranger
- From CFM 1.1.x without Ranger to CFM 2.1.5 without Ranger
- You have deployed a CDP Private Cloud Base cluster with the necessary services running (ZooKeeper, Ranger, and similar) and out-of-the-box configurations.
- You have sufficient networking connectivity and capacity between the source cluster and destination cluster.
- You have the permissions required to access and modify files on cluster nodes.
- You have reviewed the [Cloudera Flow Management Release Notes](#) and are aware of all known issues.
- You have reviewed [Apache NiFi Migration Guidance](#) and [Apache NiFi Registry Migration Guidance](#) and are aware of any differences between source and destination components.

Preserving source cluster files and directories

Learn how to locate and backup the NiFi and NiFi Registry files on the source cluster. You will need these files later in the migration process. It is also important to preserve configuration files so that you can retain any cluster customizations that you implemented on your source cluster.

Preserving custom processors/NARs

Describes the steps to preserve custom processors/NARs.

If you have created any custom NARs, a best practice is to store and reference them in a centralized location.

On each source node:

1. Create a second library directory called `custom_lib`. For example: `/opt/configuration_resources/custom_lib`.
2. Move your custom NARs to this new directory.

3. Add a new line to the `nifi.properties` file to specify this new lib directory. For example:

```
nifi.nar.library.directory.custom=/opt/configuration_resources/custom_lib
```

You will perform a similar procedure on the destination cluster nodes later in the migration process.



Note:

For HDF NARs, you may need to rebuild them to correctly depend on the CFM NARs, instead of HDF.

NiFi files to preserve

Lists the NiFi files and directories to preserve along with their default locations.

Items to preserve	Default location
The following configuration files: <ul style="list-style-type: none"> authorizers.xml bootstrap.conf bootstrap-notification-services.xml login-identity-providers.xml nifi.properties state-management.xml 	<ul style="list-style-type: none"> HDF – <code>/usr/hdf/current/nifi/conf</code> HDF or CFM standalone installations – <code><nifi-installation-directory>/conf</code>
If NiFi is secured: <ul style="list-style-type: none"> authorizations.xml users.xml 	<ul style="list-style-type: none"> HDF – <code>/var/lib/nifi/conf</code> HDF or CFM standalone installations – <code><nifi-installation-directory>/conf</code>
The following directories: <ul style="list-style-type: none"> database_repository provenance_repository local state directory 	<ul style="list-style-type: none"> HDF – <code>/var/lib/nifi</code> HDF or CFM standalone installations – <code><nifi-installation-directory>/</code>
flow.xml.gz	<ul style="list-style-type: none"> HDF – <code>/var/lib/nifi/conf</code> HDF or CFM standalone installations – <code><nifi-installation-directory>/conf</code>



Note:

It is a best practice to create NiFi content, database, flowfile and provenance repositories on separate physical disks and reference them using the directory path properties in `nifi.properties`. This best practice facilitates the migration process.

NiFi Registry files to preserve

Lists the NiFi Registry files and directories to preserve and their default locations on the source cluster.

Items to preserve	Default location
The following configuration files: <ul style="list-style-type: none"> authorizers.xml bootstrap.conf identity-providers.xml logback.xml nifi-registry.properties providers.xml 	<ul style="list-style-type: none"> HDF – <code>/usr/hdf/current/nifi-registry/conf</code> HDF or CFM standalone installations – <code><registry-installation-directory>/conf</code>
If NiFi Registry is secured: <ul style="list-style-type: none"> authorizations.xml users.xml 	<ul style="list-style-type: none"> HDF – <code>/var/lib/nifi-registry/conf</code> HDF or CFM standalone installations – <code><registry-installation-directory>/conf</code>

Items to preserve	Default location
<p>The following directories:</p> <ul style="list-style-type: none"> database <p>The <code>nifi.registry.db.url</code> property in <code>nifi-registry.properties</code> points to the location of the NiFi Registry metadata database. The default database is H2.</p> <ul style="list-style-type: none"> <code>flow_storage</code> 	<ul style="list-style-type: none"> HDF – <code>/var/lib/nifi-registry</code> HDF or CFM standalone installations – <code><registry-installation-directory>/</code>

Installing CFM 2.1.5

Provides information about installing CFM 2.1.5.

Adding and configuring the NiFi service

Provides the steps for how to add and configure your NiFi service.

Before you begin

- You have installed a CDP Private Cloud Base cluster and prepared it for the CFM deployment. For more information, see the *Preparing your CDP Private Cloud Base cluster*.
- You have equivalence between source and target clusters. For example, if your source NiFi cluster has 3 nodes, the CFM 2.1.5 NiFi cluster must have at least 3 nodes as well.
- You have reviewed the information about preserving your source cluster files and directories and made the necessary backups.

Procedure

- From Cloudera Manager, add the CFM 2.1.5 NiFi service.
- Set some initial configurations.

Generally, you can accept default values during the initial installation. However, there are some settings that you should configure before proceeding:

Property	Description
<p>Master Key Password</p> <p><code>nifi.master.key.password</code></p>	<p>This password is used when you generate the master key for sensitive properties encryption in the NiFi properties file when it is written to disk. It must contain at least 12 characters.</p>
<p>Sensitive Properties Key</p> <p><code>nifi.sensitive.props.key</code></p>	<p>This is the password used when you encrypt any sensitive property values that are configured in NiFi components. It must contain at least 12 characters.</p> <p>If you change the Sensitive Properties Key from what was used in your source cluster, you must also update the encrypted sensitive property values in the <code>flow.xml.gz</code>. Refer to the section “Migrating a Flow with Sensitive Properties” below.</p>

- Stop the NiFi service.
- Update the NiFi configuration.

In your CFM 2.1.5 NiFi, use Cloudera Manager to walk through all the configuration values and match the values from your source cluster that are not cluster specific. Examples of cluster specific values include keystore, truststore, ZooKeeper hostnames, and similar.

Reference the source NiFi configuration files collected earlier as needed. Double check all entries for typos.



Note:

Do not copy over or migrate any data repositories, local state, ZooKeeper state, or `flow.xml.gz` from your source NiFi at this time.

Example

Sample configuration changes

Update the Login Identity Provider properties.

The Template for login-identity-providers.xml from Ambari is now composed of individual properties in Cloudera Manager.

As an example, if using LDAP for authentication, the following login-identity-providers.xml:

```
<loginIdentityProviders>
  <provider>
    <identifier>ldap-provider</identifier>
    <class>org.apache.nifi.ldap.LdapProvider</class>
    <property name="Authentication Strategy">SIMPLE</property>
    <property name="Manager DN">uid=admin,ou=people,dc=hadoop,dc=
apache,dc=org</property>
    <property name="Manager Password">admin-password</property>
    <property name="Referral Strategy">FOLLOW</property>
    <property name="Connect Timeout">10 secs</property>
    <property name="Read Timeout">10 secs</property>
    <property name="Url">ldap://ctr-e144-1587379642025-3931-01-00
0003.hwx.site:33389</property>
    <property name="User Search Base">ou=people,dc=hadoop,dc=apac
he,dc=org</property>
    <property name="User Search Filter">uid={0}</property>
    <property name="Identity Strategy">USE_USERNAME</property>
    <property name="Authentication Expiration">12 hours</property>
  </provider>
</loginIdentityProviders>
```

You would use Cloudera Manager to set the following NiFi service properties instead.

- LDAP Enabled is checked
- Login Identity Provider: Default LDAP Provider Class set to org.apache.nifi.ldap.LdapProvider
- LDAP Authentication Strategy set to SIMPLE
- LDAP Manager DN set to uid=admin,ou=people,dc=hadoop,dc=apache,dc=org
- LDAP Manager Password set to admin-password
- LDAP Referral Strategy set to FOLLOW
- LDAP Connect Timeout set to 10 secs
- LDAP Read Timeout set to 10 secs
- LDAP Url set to ldap://ctr-e144-1587379642025-3931-01-000003.hwx.site:33389
- LDAP User Search Base set to ou=people,dc=hadoop,dc=apache,dc=org
- Login Identity Provider: Default LDAP User Search Filter set to uid={0}
- Login Identity Provider: Default LDAP Identity Strategy set to USE_USERNAME
- Login Identity Provider: Default LDAP Authentication Expiration set to 12 hours

There are several additional LDAP configuration requirements:

- Enable TLS/SSL for NiFi Node is checked
- Initial Admin Identity set to admin
- Login Identity Provider ID set to ldap-provider
- Authorizers: LDAP User Search Filter set to (uid=*)
- Authorizers: LDAP User Identity Attribute set to uid

What to do next

When you have completed the steps for adding and configuring the NiFi Service, you may proceed with adding and configuring the NiFi Registry service.

Related Information

[Preparing your CDP Private Cloud Base cluster](#)

[Preserving source cluster files and directories](#)

[Adding and configuring the NiFi Registry service](#)

Adding and configuring the NiFi Registry service

Provides the steps for how to add and configure your NiFi Registry service.

Before you begin

- You have installed a CDP Private Cloud Base cluster and prepared it for the CFM deployment. For more information, see the *Prepare your CDP Private Cloud Base cluster*.
- You have equivalence between source and target clusters. For example, if your source NiFi cluster has 3 nodes, the CFM 2.1.5 NiFi cluster must have at least 3 nodes as well.
- You have added the NiFi service.

Procedure

1. Add CFM 2.1.5 NiFi Registry service.
2. Set some initial configurations.

Generally, you can accept default values during the initial installation. However, there are some settings that you should configure before proceeding:

Property	Description
Master Key Password <code>nifi.registry.master.key.password</code>	This password is used to generate the master key for encrypting NiFi Registry properties on the filesystem.

3. Stop the NiFi Registry service.
4. Update the NiFi Registry configuration.

In your CFM 2.1.5 NiFi Registry, use Cloudera Manager to walk through all the configuration values and match the values from your source cluster that are not cluster specific. Examples of cluster specific values include keystore, truststore, and similar.

Reference the source NiFi Registry configuration files collected earlier as needed. Double check all entries for typos.



Note:

Do not copy over or migrate any data repositories, local state, ZooKeeper state, or flow.xml.gz from your source NiFi Registry at this time.

Example

Sample configuration changes

Update the Login Identity Provider properties.

The Template for identity-providers.xml from Ambari is now composed of individual properties in Cloudera Manager.

As an example, if using LDAP for authentication, the following identity-providers.xml:

```
<identityProviders>
  <provider>
    <identifier>ldap-provider</identifier>
    <class>org.apache.nifi.registry.security.ldap.LdapIdentityProvider</class>
    <property name="Authentication Strategy">SIMPLE</property>
  </provider>
</identityProviders>
```

```

    <property name="Manager DN">uid=admin,ou=people,dc=hadoop,dc=apache
,dc=org</property>
    <property name="Manager Password">admin-password</property>
    <property name="Referral Strategy">FOLLOW</property>
    <property name="Connect Timeout">10 secs</property>
    <property name="Read Timeout">10 secs</property>
    <property name="Url">ldap://ctr-e144-1587379642025-3931-01-000003.h
wx.site:33389</property>
    <property name="User Search Base">ou=people,dc=hadoop,dc=apache,dc=
org</property>
    <property name="User Search Filter">uid={0}</property>
    <property name="Identity Strategy">USE_USERNAME</property>
    <property name="Authentication Expiration">12 hours</property>
  </provider>
</identityProviders>

```

You would use Cloudera Manager to set the following NiFi Registry service properties instead.

- LDAP Enabled is checked
- Identity Provider: Default LDAP Provider Class set to org.apache.nifi.registry.security.ldap.LdapIdentityProvider
- LDAP Authentication Strategy set to SIMPLE
- LDAP Manager DN set to uid=admin,ou=people,dc=hadoop,dc=apache,dc=org
- LDAP Manager Password set to admin-password
- LDAP Referral Strategy set to FOLLOW
- LDAP Connect Timeout set to 10 secs
- LDAP Read Timeout set to 10 secs
- LDAP Url set to ldap://ctr-e144-1587379642025-3931-01-000003.hwx.site:33389
- LDAP User Search Base set to ou=people,dc=hadoop,dc=apache,dc=org
- Identity Provider: Default LDAP User Search Filter set to uid={0}
- Identity Provider: Default LDAP Identity Strategy set to USE_USERNAME
- Identity Provider: Default LDAP Authentication Expiration set to 12 hours

There are several additional LDAP configuration requirements:

- Enable TLS/SSL for NiFi Registry is checked
- Initial Admin Identity set to admin
- Identity Provider Identifier set to ldap-provider
- Authorizers: LDAP User Search Filter set to (uid=*)
- Authorizers: LDAP User Identity Attribute set to uid
- Client Authentication Required is unchecked

What to do next

When you have finished adding and configuring both the NiFi and NiFi Registry services, verify your CFM 2.1.5 installation.

Related Information

[Preparing your CDP Private Cloud Base cluster](#)

[Adding and configuring the NiFi service](#)

[Verifying CFM 2.1.5](#)

Verifying CFM 2.1.5

Before continuing the upgrade process by migrating NiFi and NiFi Registry data, you need to verify your CFM 2.1.5 installation.

Before you begin

If you updated the Initial Admin Identity (`nifi.initial.admin.identity`) after the initial NiFi service installation, select Actions | Reset File-based Authorizer Users and Policies before you start the NiFi service. This archives the existing `users.xml` and `authorizations.xml` files and new ones will be generated based on configuration changes. Similarly, if you updated the Initial Admin Identity (`nifi.registry.initial.admin.identity`) after the initial NiFi Registry service installation, select Actions | Reset File-based Authorizer Users and Policies before you start the NiFi Registry service.

Procedure

1. Start the NiFi service.
2. Start the NiFi Registry service.
3. Verify NiFi and NiFi Registry started successfully securely over HTTPS.
4. Log in to NiFi using your authorized admin user.

You should see a blank canvas.

5. Log in to NiFi Registry using your authorized admin user.

You will see no resources.



Tip:

Note the Registry URL so that it can be referenced later.

6. Once you are satisfied that all services are working correctly, shutdown both NiFi and NiFi Registry services and continue with the migration.
7. Remove `flow.xml.gz` and `flow.json.gz` generated by CFM 2.1.4 NiFi during verification startup. (Default location is `/var/lib/nifi/`.)

What to do next



Important:

The following sections may incur downtime. Read all sections before proceeding. A migration without downtime is possible depending on the use cases involved. Contact your Cloudera representative to engage Professional Services for guidance.

When you are ready to proceed, your next step is to shut down the source services.

Related Information

[Adding and configuring the NiFi Registry service](#)

[Clearing activity and shutting down source services](#)

Clearing activity and shutting down source services

Before continuing the upgrade process by migrating NiFi and NiFi Registry data, you need to clear the activity and shut down your source cluster NiFi and NiFi Registry services.

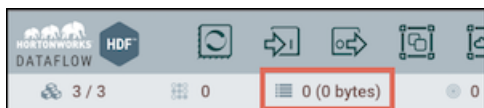
Before you begin

- You have preserved your source cluster files and directories.
- You have installed and verified CFM 2.1.5.

Procedure

1. In the source NiFi cluster, stop all the source processors to prevent the ingestion of new data.

2. Confirm that there is no active data in any of the queues by monitoring the left side of the NiFi status bar. When all active data has stopped, the status bar shows 0 FlowFiles and 0 bytes of data:



3. Shut down the source NiFi service.
4. Shut down the source NiFi Registry service.

What to do next

When you have finished shutting down your source services, follow the steps to migrate the NiFi data directories.

Related Information

[Verifying CFM 2.1.5](#)

[Migrating the NiFi data directories](#)

Migrating the NiFi data directories

The first stage of migrating NiFi data is the migration of NiFi data directories.

Before you begin

When you are migrating any of the following from the source cluster to the destination cluster, ensure that the file and directory ownerships and permissions are consistent.

About this task

The database_repository consists of two H2 databases and their corresponding lock files:

- nifi-flow-audit.h2.db
- nifi-flow-audit.lock.db
- nifi-user-keys.h2.db
- nifi-user-keys.lock.db

The nifi-user-keys.h2.db contains information about who has logged into NiFi, if NiFi has been secured. The nifi-flow-audit.h2.db contains flow configuration history. These databases only need to be migrated if this historical information needs to be retained, but note that any NiFi nodes referenced in them will refer to the source nodes. If you do not wish to retain the database information, you may skip step 2.

Procedure

1. Copy the entire provenance_repository directory from a source cluster node to a node on the destination cluster. Provenance repositories cannot be merged. For example: source Node 1 # destination Node 1, source Node 2 # destination Node 2, ... source Node N # destination Node N.



Note:

The path to the destination provenance repository can be the same as the source provenance repository (if on a separate physical disk, for example), but it is not a requirement. You must place the provenance_repository directory in the path assigned to the nifi.provenance.repository.directory property in the destination cluster. The default path is /var/lib/nifi/provenance_repository.

2. Copy the entire `database_repository` directory from a source cluster node to a node on the destination cluster. For example: source Node 1 # destination Node 1, source Node 2 # destination Node 2, ... source Node N # destination Node N.

**Note:**

The path to the destination database repository can be the same as the source database repository (if on a separate physical disk, for example), but it is not a requirement. You must place the `database_repository` directory in the path assigned to the `nifi.database.directory` property in the destination cluster. The default path is `/var/lib/nifi/database_repository`.

What to do next

When you have completed the `provenance_repository` and `database_repository` migration, proceed by migrating the NiFi `flow.xml.gz` file.

Related Information

[Clearing activity and shutting down source services](#)

[Migrating the NiFi `flow.xml.gz` file](#)

Migrating the NiFi `flow.xml.gz` file

Provides steps for migrating the NiFi `flow.xml.gz` file.

Before you begin

When you migrate the `flow.xml.gz` from the source cluster to the destination cluster, ensure that the file ownership and permission are consistent.

Copy the `flow.xml.gz` file from any node in the source cluster to all nodes on the destination cluster. The default CFM 2.1.5 location is the `var/lib/nifi` directory.

Before you copy the `flow.xml.gz` file to the destination cluster, you must edit it to remove any references to the source cluster. The content of the `flow.xml.gz` file is different, based on source cluster activities. Here are some examples of common edits you must make.

Updating encryption algorithm for sensitive properties

You must update the encryption algorithm for sensitive properties in the `flow.xml.gz` file.

About this task

Starting with Cloudera Flow Management 2.1.5, the default algorithm for encrypting the sensitive properties in the `flow.xml.gz` file changed to `NIFI_PBKDF2_AES_GCM_256` to provide greater security.

To update the `flow.xml.gz` file with this new algorithm, execute the following command:

```
/opt/cloudera/parcels/CFM-2.1.5.0-215/TOOLKIT/bin/encrypt-config.sh -n <nifi
.properties from original NiFi>
-f <flow.xml.gz from original NiFi> -x -s <sensitive props key from new NiFi>
-i> -b <bootstrap.conf from original NiFi> -A NIFI_PBKDF2_AES_GCM_256
-g <new flow.xml.gz filename>
```

What to do next

When you made all the necessary edits, you can deploy the new `flow.xml.gz` file across the NiFi nodes.

Removing unnecessary reporting tasks

You must remove the unnecessary reporting tasks from the `flow.xml.gz` file.

About this task

You can remove the `AmbariReportingTask` used by your HDF NiFi cluster, as it is not used by Cloudera Manager.

Procedure

1. Unzip `flow.xml.gz`.
2. Edit the resulting `flow.xml` file by removing the reporting task:

```
<reportingTask>
  <id>3b80ba0f-a6c0-48db-b721-4dbc04cef28e</id>
  <name>AmbariReportingTask</name>
  <comment/>
  <class>org.apache.nifi.reporting.ambari.AmbariReportingTask</class>
  <bundle>
    <group>org.apache.nifi</group>
    <artifact>nifi-ambari-nar</artifact>
    <version>1.11.4.3.5.1.0-17</version>
  </bundle>
  <schedulingPeriod>1 mins</schedulingPeriod>
  <scheduledState>RUNNING</scheduledState>
  <schedulingStrategy>TIMER_DRIVEN</schedulingStrategy>
  <property>
    <name>Metrics Collector URL</name>
    <value>${ambari.metrics.collector.url}</value>
  </property>
  <property>
    <name>Application ID</name>
    <value>${ambari.application.id}</value>
  </property>
  <property>
    <name>Hostname</name>
    <value>${hostname(true)}</value>
  </property>
  <property>
    <name>Process Group ID</name>
  </property>
</reportingTask>
```



Note:

Reporting tasks are bookended by the tags `<reportingTasks>` and `<reportingTasks/>`. If you no longer have any reporting tasks in your flow, you should still retain the closing tag `<reportingTasks/>` in the `flow.xml` file.

3. Save your changes.
4. Zip the `flow.xml` file:

```
gzip flow.xml
```

What to do next

When you have completed the steps to remove unnecessary reporting tasks, proceed by updating the Registry Client.

Related Information

[Migrating the NiFi data directories](#)

[Updating the Registry Client](#)

Updating the Registry Client

You must update the Registry Client information in the flow.xml.gz file.

Before you begin

You have removed any unnecessary reporting tasks.

Procedure

1. Unzip flow.xml.gz.
2. Edit the resulting flow.xml file by replacing the source Registry URL with the destination Registry URL and change the Registry Name if desired:

```
<registries>
  <flowRegistry>
    <id>95503839-0172-1000-ffff-ffffe2b7b914</id>
    <name>CFM Registry</name>
    <url>https://cfm_registry_node:61443</url>
    <description/>
  </flowRegistry>
</registries>
```

3. Save your changes.
4. Zip the flow.xml file:

```
gzip flow.xml
```

What to do next

When you have updated the Registry Client, proceed by updating references to source cluster nodes with destination cluster nodes.

Related Information

[Removing unnecessary reporting tasks](#)

[Updating references to cluster nodes](#)

Updating references to cluster nodes

You must update the references to source cluster nodes with destination cluster nodes in the flow.xml.gz file.

Before you begin

You have removed unnecessary reporting tasks and updated the Registry Client information.

Procedure

1. Unzip flow.xml.gz.
2. Search and replace URLs that refer to the source NiFi nodes with the equivalent destination NiFi nodes. For example, in a Remote Process Group:

```
<remoteProcessGroup>
  <id>c8647bee-0172-1000-0000-00001ff1d10f</id>
  <name>NiFi Flow</name>
  <position x="968.0" y="520.0"/>
  <comment/>
  <url>https://remote_instance_host:8443/nifi</url>
  <urls>https://remote_instance_host:8443/nifi</urls>
```

```

<timeout>30 sec</timeout>
<yieldPeriod>10 sec</yieldPeriod>
<transmitting>false</transmitting>
<transportProtocol>HTTP</transportProtocol>
<proxyHost/>
<proxyUser/>
<inputPort>
  <id>6ade8167-dc3c-3afc-b22e-f38465fdf601</id>
  <name>File Listing</name>
  <position x="0.0" y="0.0"/>
  <comments/>
  <scheduledState>STOPPED</scheduledState>
  <targetId>c85b69ba-0172-1000-0000-000077bde971</targetId>
  <maxConcurrentTasks>1</maxConcurrentTasks>
  <useCompression>false</useCompression>
  <batchCount>1</batchCount>
</inputPort>
</remoteProcessGroup>

```

3. Save your changes.
4. Zip the flow.xml file:

```
gzip flow.xml
```

What to do next

When you have finished updating the cluster node references, proceed by updating a flow with sensitive properties.

Related Information

[Updating the Registry Client](#)

[Updating a flow with sensitive properties](#)

Updating a flow with sensitive properties

If the Sensitive Properties Key (nifi.sensitive.props.key) is changing from the source cluster to the destination cluster, you must update the flow.xml.gz file prior to copying it to each node.

When a value is set for nifi.sensitive.props.key, the specified key is used to encrypt sensitive properties in the flow (password fields in components for example). You can use the Encrypt-Config tool in the NiFi Toolkit to migrate the key and update the flow.xml.gz. Encrypt-Config performs the following actions:

- Reads the existing flow.xml.gz and decrypts the sensitive values using the current key.
- Encrypts all the sensitive values with a specified new key.
- Updates the existing nifi.properties and flow.xml.gz files or creates new versions of them.

See *Using the Apache NiFi Toolkit* for complete information on Encrypt-Config.



Note:

In an HDF cluster, the NiFi Toolkit scripts are located in /usr/hdf/current/nifi-toolkit/bin.

Here is an example Encrypt-Config tool command:

```

$ ./nifi-toolkit-<version>/bin/encrypt-config.sh
-f /path/to/nifi_source/flow.xml.gz
-g /path/to/create/updated/flow.xml.gz
-s <new-password>
-n /path/to/nifi_source/nifi.properties
-o /path/to/create/updated/nifi.properties
-x

```


Where:

- -f specifies the source flow.xml.gz
- -g specifies the destination flow.xml.gz
- -s specifies the new sensitive properties key
- -n specifies the source nifi.properties
- -o specifies the destination nifi.properties
- -x tells the Encrypt-Config tool to only process the sensitive properties

If values in nifi.properties have been encrypted using the Encrypt Configuration Master Key Password property in Ambari (equivalent to the nifi.master.key.password property in CFM), add the -b option:

```
$ ./nifi-toolkit-<version>/bin/encrypt-config.sh
-b /path/to/nifi_source/bootstrap.conf
-f /path/to/nifi_source/flow.xml.gz
-g /path/to/create/updated/flow.xml.gz
-s <new-password>
-n /path/to/nifi_source/nifi.properties
-o /path/to/create/updated/nifi.properties
-x
```

Where:

- -b specifies the source NiFi bootstrap.conf

Related Information

[Using the Apache NiFi Toolkit](#)

[Updating references to cluster nodes](#)

Migrating authorization policies

Describes how to migrate both Ranger and file-based policies for NiFi and NiFi Registry.

Migrating NiFi Ranger-based policies

Provides the steps for migrating NiFi Ranger-based policies.

Before you begin

- You have installed the Ranger service on your destination cluster.
- You have selected Ranger as a NiFi dependency.

About this task

If the source cluster uses Ranger policies for NiFi authorizations and you require the same Ranger policies on the destination cluster, migrate the existing Ranger policies using the Ranger Import/Export feature.

Procedure

1. In your source Ranger UI, select Access Manager | Resource Based Policies. On the Service Manager page, select Export. Remove all services listed except NiFi and select Export. A JSON file is exported.
2. In your destination Ranger UI, select Access Manager | Resource Based Policies. On the Service Manager page, select the NiFi service. Delete all of the existing policies on the service, being careful not to delete the NiFi service.
3. Return to the Service Manager page in your destination Ranger. Select Import. Select the source JSON file you exported in Step 1. Map the source NiFi Ranger service to the destination NiFi Ranger service. Select Import.

4. For NiFi service policies where source NiFi nodes are referenced (for example, Proxy policy), add the group nifi to those conditions, then delete the source nodes from those policies.
5. Edit the users.xml from the source cluster by removing source node users. Replace the users.xml on each destination cluster NiFi node with the modified users.xml. The default CFM 2.1.5 location is /var/lib/nifi.

What to do next

When you have finished migrating NiFi Ranger-based policies, proceed with the steps for migrating NiFi Registry Ranger-based policies.

Related Information

[Installing Cloudera Manager and a CDP Private Cloud Base cluster](#)

[Migrating NiFi Registry Ranger-based policies](#)

Migrating NiFi Registry Ranger-based policies

Provides the steps for migrating NiFi Registry Ranger-based policies.

About this task

If the source cluster uses Ranger policies for NiFi Registry authorizations and you require the same Ranger policies on the destination cluster, migrate the existing Ranger policies using the Ranger Import/Export feature.

Before you begin

- You have installed the Ranger service on your destination cluster.
- You have selected Ranger as a NiFi Registry dependency.

Procedure

1. In your source Ranger UI, select Access Manager | Resource Based Policies. On the Service Manager page, select Export. Remove all services listed except NiFi Registry and select Export. A JSON file is exported.
2. In your destination Ranger UI, select Access Manager | Resource Based Policies. On the Service Manager page, select the NiFi Registry Ranger service. Delete all of the existing policies on the service, being careful not to delete the NiFi Registry service.
3. Return to the Service Manager page in your destination Ranger. Select Import. Select the source JSON file you exported in Step 1. Map the source NiFi Registry Ranger service to the destination NiFi Registry Ranger service. Select Import.
4. For NiFi Registry service policies where source NiFi nodes are referenced (for example, Proxy and Bucket policies), add the group nifiregistry to those conditions, then delete the source nodes from those policies.
5. Edit the users.xml from the source cluster by removing source node users. Replace the users.xml on the destination cluster NiFi Registry node with the modified users.xml. The default CFM 2.1.5 location is /var/lib/nifiregistry.

What to do next

When you have completed migrating NiFi Registry policies, you may proceed to migrating NiFi state and custom components.

Related Information

[Installing Cloudera Manager and a CDP Private Cloud Base cluster](#)

[Migrating NiFi Ranger-based policies](#)

[Migrating NiFi file-based policies](#)

Migrating NiFi file-based policies

Provides information about and examples of migrating NiFi file-based policies.

To migrate NiFi file-based authorization policies, you will perform the following edits.

Edit the `users.xml` from the source cluster by removing references to the source node users. Replace the `users.xml` on each destination NiFi node with the modified `users.xml`. The default CFM 2.1.5 location is `/var/lib/nifi`.

CFM NiFi uses the `CMUserGroupProvider`, configured in the `authorizers.xml` file, and places all the NiFi node hostnames in the `nifi` group. Edit the `authorizations.xml` from the source cluster by removing source node users from each policy they were assigned and replacing them with the `nifi` group identifier.

For example, if HDF NiFi had three NiFi node users on the “proxy user requests” policy:

```
<policy identifier="287edf48-da72-359b-8f61-da5d4c45a270" resource="/proxy"
  action="W">
  <user identifier="fd4f71d3-7b7c-3286-bec0-d42a752f1e0c"/>
  <user identifier="80ba71da-7789-3c7b-924f-041d598c8137"/>
  <user identifier="e32ed0da-e652-39e6-86db-48cfa3750a9f"/>
</policy>
```

You should edit the policy with the following information for CFM 2.1.5:

```
<policy identifier="287edf48-da72-359b-8f61-da5d4c45a270" resource="/proxy"
  action="W">
  <group identifier="nifi"/>
</policy>
```

You must make similar changes if your flow uses Site-to-Site. You must update the “retrieve site-to-site details” global access policy:

```
<policy identifier="c86712ce-0172-1000-0000-00007a4ea450" resource="/site-to-
-site" action="R">
  <group identifier="nifi"/>
</policy>
```

You must also update the “retrieve data via site-to-site” policy on related input ports:

```
<policy identifier="c8a608dd-0172-1000-ffff-ffffec1193ac" resource="/data-t
ransfer/input-ports/c85b69ba-0172-1000-0000-000077bde971" action="W">
  <group identifier="nifi"/>
</policy>
```



Note:

For any policies that require both group and user identifiers, the group identifier must be placed before the user identifier on each policy.

Replace the `authorizations.xml` on each destination cluster NiFi node with the modified `authorizations.xml`. The default CFM 2.1.5 location is `/var/lib/nifi`.

After you finish

When you have finished migrating NiFi file-based policies, proceed with the steps for migrating NiFi Registry file-based policies.

Related Information

[Installing Cloudera Manager and a CDP Private Cloud Base cluster](#)

[Migrating NiFi Registry Ranger-based policies](#)

[Migrating NiFi Registry file-based policies](#)

Migrating NiFi Registry file-based policies

Provides information about and examples of migrating NiFi Registry file-based policies.

Edit the users.xml from the source cluster by removing source node users. Replace the users.xml on the NiFi Registry node with the modified users.xml. The default CFM 2.1.5 location is /var/lib/nifiregistry.

CFM NiFi Registry uses the CMUserGroupProvider, configured in the authorizers.xml file, and places all the NiFi node hostnames in the nifiregistry group. Edit the authorizations.xml from the source cluster by removing source node users from each policy they were assigned and replacing them with the nifiregistry group identifier.

For example, if HDF NiFi Registry had three NiFi node users on each of the Read/Write/Delete proxy policies:

```
<policy identifier="d59a54f7-6dd6-34ad-a279-a26ffdb9eef8" resource="/proxy"
  action="R">
  <user identifier="21232f29-7a57-35a7-8389-4a0e4a801fc3"/>
  <user identifier="fd4f71d3-7b7c-3286-bec0-d42a752f1e0c"/>
  <user identifier="80ba71da-7789-3c7b-924f-041d598c8137"/>
  <user identifier="e32ed0da-e652-39e6-86db-48cfa3750a9f"/>
</policy>
<policy identifier="287edf48-da72-359b-8f61-da5d4c45a270" resource="/proxy"
  action="W">
  <user identifier="21232f29-7a57-35a7-8389-4a0e4a801fc3"/>
  <user identifier="fd4f71d3-7b7c-3286-bec0-d42a752f1e0c"/>
  <user identifier="80ba71da-7789-3c7b-924f-041d598c8137"/>
  <user identifier="e32ed0da-e652-39e6-86db-48cfa3750a9f"/>
</policy>
<policy identifier="6dbdbbfd-8a7d-32e1-ba3e-f600e6c69791" resource="/proxy"
  action="D">
  <user identifier="21232f29-7a57-35a7-8389-4a0e4a801fc3"/>
  <user identifier="fd4f71d3-7b7c-3286-bec0-d42a752f1e0c"/>
  <user identifier="80ba71da-7789-3c7b-924f-041d598c8137"/>
  <user identifier="e32ed0da-e652-39e6-86db-48cfa3750a9f"/>
</policy>
```

You should edit these policies with the following information for CFM 2.1.5:

```
<policy identifier="d59a54f7-6dd6-34ad-a279-a26ffdb9eef8" resource="/proxy"
  action="R">
  <group identifier="nifiregistry"/>
  <user identifier="21232f29-7a57-35a7-8389-4a0e4a801fc3"/>
</policy>
<policy identifier="287edf48-da72-359b-8f61-da5d4c45a270" resource="/proxy"
  action="W">
  <group identifier="nifiregistry"/>
  <user identifier="21232f29-7a57-35a7-8389-4a0e4a801fc3"/>
</policy>
<policy identifier="6dbdbbfd-8a7d-32e1-ba3e-f600e6c69791" resource="/proxy"
  action="D">
  <group identifier="nifiregistry"/>
  <user identifier="21232f29-7a57-35a7-8389-4a0e4a801fc3"/>
</policy>
```



Note:

For any policies that require both group and user identifiers, the group identifier must be placed before the user identifier on each policy.

Replace the authorizations.xml on the NiFi Registry node with the modified authorizations.xml. The default CFM 2.1.5 location is /var/lib/nifiregistry.

After you finish

When you have completed migrating NiFi Registry file-based policies, you may proceed to migrating NiFi state and custom components.

Related Information

[Migrating NiFi file-based policies](#)

[Migrating NiFi state and custom components](#)

Migrating NiFi state and custom components

Provides steps for migrating NiFi state and custom components.

Copy local state

The state-management.xml file contains a local-provider value with a directory path. Copy the contents of local state directory from each source NiFi node to a node in the destination cluster. For example: source Node 1 # destination Node 1, source Node 2 # destination Node 2, ..., source Node N # destination Node N.

The default state directory path in CFM 2.1.5 is /var/lib/nifi/state.

Copy cluster state

The state-management.xml file contains a cluster-provider value with ZooKeeper configuration. Use the ZooKeeper Migrator in the NiFi Toolkit to migrate NiFi ZooKeeper content from the source cluster to the destination.



Note:

In an HDF cluster, the NiFi Toolkit scripts are located in /usr/hdf/current/nifi-toolkit/bin.

1. Export the NiFi component data from the source ZooKeeper:

```
./zk-migrator.sh
-r
-z sourceHostname:sourceClientPort/sourceRootPath/components
-f /path/to/export/zk-source-data.json
```



Note:

Ensure that the export directory exists as it is not created by the command.

2. Migrate the source ZooKeeper data to the destination ZooKeeper:

```
./zk-migrator.sh
-s
-z destinationHostname:destinationClientPort/destinationRootPath/compon
ents
-f /path/to/export/zk-source-data.json
```



Note:

In CFM 2.1.5, the NiFi Toolkit scripts are located in /opt/cloudera/parcels/CFM-<version>/TOOLKIT/bin.

For more information on the ZooKeeper Migrator, see *Using the Apache NiFi Toolkit*.

Manage custom components

1. In Cloudera Manager:
 - a. Find the NiFi configuration property “NiFi Node Advanced Configuration Snippet (Safety Valve) for staging/nifi.properties.xml”
 - b. Click “+” to add a snippet.
 - c. For the Name, enter: `nifi.nar.library.directory.custom`
 - d. For the Value, enter: `/opt/configuration_resources/custom_lib`
 - e. Enter a description.

For example:

NiFi Node Advanced Configuration Snippet (Safety Valve) for staging/nifi.properties.xml

NiFi Node Default Group

View as XML

Name:

Value:

Description:

☐ Final



Tip:

To specify that the property values cannot be overridden, select the Final checkbox.

2. On every destination NiFi cluster node, copy any custom NiFi NARs to the directory path specified by the Value field.

After you finish

When you have finished migrating NiFi state and any custom components, proceed by migrating the NiFi Registry data storage.

Related Information

[Using the Apache NiFi Toolkit](#)

[Migrating NiFi Registry file-based policies](#)

[Migrating NiFi Registry data storage](#)

Migrating NiFi Registry data storage

Provides information for migrating your NiFi Registry metadata database, your flow storage, and your bundle storage configurations.

Migrating the metadata database

Provides steps for migrating the H2, PostgreSQL, or MySQL metadata database.

H2 database (default)

1. Confirm the location of the source Registry metadata database specified by the `nifi.registry.db.url` property in `nifi-registry.properties`.
2. Copy the database from the source NiFi Registry to the location specified by the NiFi Registry JDBC Url (`nifi.registry.db.url`) in the destination NiFi Registry configuration. The default directory path in CFM 2.1.5 is `/var/lib/nifi/registry/database`.



Note:

The NiFi Registry Database Password (`nifi.registry.db.password`) in CFM must match the password used with the source H2 database.

PostgreSQL

1. In the destination Registry, match the configuration in the source `nifi-registry.properties`. Sample properties:

- NiFi Registry JDBC Url (`nifi.registry.db.url`) – `jdbc:postgresql://<POSTGRES-HOSTNAME>/nifireg`
- NiFi Registry JDBC Driver (`nifi.registry.db.driver.class`) – `org.postgresql.Driver`
- NiFi Registry H2 directory storage location (`nifi.registry.db.driver.directory`) – `/path/to/drivers`



Note:

The NiFi Registry H2 directory storage location specifies the NiFi Registry database driver directory. The H2 database is used by default. Update this field when you are configuring it for an external database.

- Username for NiFi Registry metadata database (`nifi.registry.db.username`) – `nifireg`
- Password for NiFi Registry metadata database (`nifi.registry.db.password`) – `changeme`

2. Save the changes.

3. Download the Postgres JDBC driver and place it in the expected driver directory:

```
/path/to/drivers/postgresql-driver.jar
```



Note:

These steps assume the destination registry is pointing to the same external database referenced by the source registry. If the source registry has new databases, you must migrate the data from the source databases by following the database specific steps for export and import.

MySQL



Note: These steps assume the destination registry is pointing to the same external database referenced by the source registry. If the source registry has new databases, you must migrate the data from the source databases by following the database specific steps for export and import.

1. In the destination Registry, match the configuration in the source `nifi-registry.properties`. Sample properties:

- NiFi Registry JDBC Url (`nifi.registry.db.url`) – `jdbc:mysql://<MYSQL-HOSTNAME>/nifi_registry`
- NiFi Registry JDBC Driver (`nifi.registry.db.driver.class`) – `com.mysql.cj.jdbc.Driver`
- NiFi Registry H2 directory storage location (`nifi.registry.db.driver.directory`) – `/path/to/drivers`



Note:

The NiFi Registry H2 directory storage location specifies the NiFi Registry database driver directory. The H2 database is used by default. Update this field when you are configuring it for an external database.

- Username for NiFi Registry metadata database (`nifi.registry.db.username`) – `nifireg`
- Password for NiFi Registry metadata database (`nifi.registry.db.password`) – `changeme`

2. Save the changes.

3. Download the MySQL JDBC driver and place it in the expected driver directory:

```
/path/to/drivers/mysql-connector-java-driver.jar
```

After you finish

When you have completed your metadata database migration, you may proceed by migrating your flow storage.

Related Information

[Migrating flow storage](#)

Migrating flow storage

Provides steps for migrating your local file system, git, or database table flow storage.

Local file system flow storage (FileSystemFlowPersistenceProvider)

This provider is the default Flow Persistence Provider.

1. Confirm the configuration of the flow storage directory specified in the source registry providers.xml:

```
<flowPersistenceProvider>
<class>org.apache.nifi.registry.provider.flow.FileSystemFlowPersistencePr
ovider</class>
<property name="Flow Storage Directory">./flow_storage</property>
</flowPersistenceProvider>
```

2. Copy the flow storage directory from the source registry installation to the location specified by the Providers: Default Flow Persistence File Provider Property - Flow Storage Directory (xml.providers.flowPersistenceProvider.file-provider.property.Flow Storage Directory) configuration property in the destination registry. The default directory path in CFM 2.1.5 is /var/lib/nifi/registry/flow_storage.

Git flow storage (GitFlowPersistenceProvider)

This provider stores flow contents under a Git directory.

1. Confirm the configuration of the flow storage directory specified in the source registry providers.xml:

```
<flowPersistenceProvider>
<class>org.apache.nifi.registry.provider.flow.git.GitFlowPersistenceProvi
der</class>
  <property name="Flow Storage Directory">./your_repo</property>
  <property name="Remote To Push">origin</property>
  <property name="Remote Access User">git_user</property>
  <property name="Remote Access Password">git_password</property>
</flowPersistenceProvider>
```

2. In the git repository directory, run git remote -v to show the URL of the remote.
3. Clone the git repository in the destination environment.
4. Configure the destination registry to point at the git repo. In Cloudera Manager:
 - Uncheck Providers: Enable File Flow Persistence Provider (xml.providers.flowPersistenceProvider.file-provider.enabled) so that FileSystemFlowPersistenceProvider is no longer enabled
 - Find the NiFi Registry configuration property NiFi Registry Advanced Configuration Snippet (Safety Valve) for staging/providers.xml. Add the following snippets:
 - Name:

```
xml.providers.flowPersistenceProvider.git-provider.enabled
```

Value: true

- Name:

```
xml.providers.flowPersistenceProvider.git-provider.class
```

Value: org.apache.nifi.registry.provider.flow.git.GitFlowPersistenceProvider

- Name:

```
xml.providers.flowPersistenceProvider.git-provider.property.Flow Sto
rage
```


Directory

Value: ./your_repo

- Name:

```
xml.providers.flowPersistenceProvider.git-provider.property.Remote To
Push
```

Value: origin

- Name:

```
xml.providers.flowPersistenceProvider.git-provider.property.Remote A
ccess
User
```

Value: git_user

- Name:

```
xml.providers.flowPersistenceProvider.git-provider.property.Remote A
ccess
Password
```

Value: git_password

5. Save the changes.

**Note:**

If “Remote To Push” is not defined and flows are stored in a local git repo, copy the flow storage directory from the source NiFi Registry installation to the referenced location in the destination NiFi Registry.

Database table storage (DatabaseFlowPersistenceProvider)

This provider leverages the same database used for the metadata database.

1. Confirm the configuration of the flow storage directory specified in the source registry providers.xml:

```
<flowPersistenceProvider>
<class>org.apache.nifi.registry.provider.flow.DatabaseFlowPersistenceProv
ider</class>
</flowPersistenceProvider>
```

2. Configure the destination registry. In Cloudera Manager:

- Uncheck Providers: Enable File Flow Persistence Provider (xml.providers.flowPersistenceProvider.file-provider.enabled) so that FileSystemFlowPersistenceProvider is no longer enabled
- Find the NiFi Registry configuration property NiFi Registry Advanced Configuration Snippet (Safety Valve) for staging/providers.xml. Add the following snippets:

- Name:

```
xml.providers.flowPersistenceProvider.database-provider.enabled
```

Value: true

- Name:

```
xml.providers.flowPersistenceProvider.database-provider.class
```

Value: org.apache.nifi.registry.provider.flow.DatabaseFlowPersistenceProvider

3. Save the changes.

**Note:**

If the versioned flows have nested version controlled process groups, you must update the registry URL in the stored flows, as they reference the source NiFi Registry. For example, a flow snapshot file would have this sample content:

```
"versionedFlowCoordinates" : {
  "bucketId" : "4d00e96b-dd33-447a-81bf-a240832e3272",
  "flowId" : "873ef981-48c1-41c2-9c58-89c6ae93d891",
  "registryUrl" : "https://hdf_registry_hostname:61080",
  "version" : 1
}
```

Replace the HDF Registry URL with the CFM Registry URL in each of the flow snapshot files.

After you finish

When you have completed the flow storage migration, you may proceed by migrating your bundle storage configurations

Related Information

[Migrating the metadata database](#)

[Migrating bundle storage configurations](#)

Migrating bundle storage configurations

Provides steps on how to migrate your local file system or AWS S3 bucket bundle storage configurations.

Local file system bundle storage (FileSystemBundlePersistenceProvider)

This is the default Bundle Persistence Provider.

1. Confirm the location of the bundle storage directory specified in the source registry providers.xml:

```
<extensionBundlePersistenceProvider>
<class>org.apache.nifi.registry.provider.extension.FileSystemBundlePersistenceProvider</class>
<property name="Extension Bundle Storage Directory">/var/lib/nifi-registry/extension_bundles</property>
</extensionBundlePersistenceProvider>
```

2. Copy the bundle storage directory from the source Registry installation to the location specified by the Providers: Default Extension Bundle Persistence File Provider Property - Extension Bundle Storage Directory (xml.providers.extensionBundlePersistenceProvider.file-bundle-provider.property.Extension Bundle Storage Directory) configuration property in the destination registry. The default CFM 2.1.5 location is /var/lib/nifi-registry/extension_bundles.

AWS S3 bucket bundle storage (S3BundlePersistenceProvider)

This provider stores the content of extension bundles in a AWS S3 bucket.

1. Confirm the location of the bundle storage directory specified in the source registry providers.xml:

```
<extensionBundlePersistenceProvider>
<class>org.apache.nifi.registry.provider.extension.S3BundlePersistenceProvider</class>
<property name="Region">us-east-1</property>
<property name="Bucket Name">my-bundles</property>
<property name="Key Prefix"></property>
<property name="Credentials Provider">DEFAULT_CHAIN</property>
```

```
<property name="Access Key">*****</property>
<property name="Secret Access Key">*****</property>
<property name="Endpoint URL"></property>
</extensionBundlePersistenceProvider>
```

2. Depending on your source credential provider (assuming the "default chain" which checks system properties, environment variables, and credential profiles), set up the credential provider on the destination system similarly. If you are using the "static" provider, specify the Access Key and Secret Access Key. In Cloudera Manager:

- Uncheck Providers: Enable File Flow Persistence Provider (xml.providers.flowPersistenceProvider.file-provider.enabled) so that FileSystemFlowPersistenceProvider is no longer enabled
- Find the NiFi Registry configuration property NiFi Registry Advanced Configuration Snippet (Safety Valve) for staging/providers.xml. Add the following snippets:

- Name:

```
xml.providers.extensionBundlePersistenceProvider.s3-bundle-provider.
enabled
```

Value: true

- Name:

```
xml.providers.extensionBundlePersistenceProvider.s3-bundle-provider.
class
```

Value: org.apache.nifi.registry.provider.extension.S3BundlePersistenceProvider

- Name:

```
xml.providers.extensionBundlePersistenceProvider.s3-bundle-provider.
property.Region
```

Value: us-east-1

- Name:

```
xml.providers.extensionBundlePersistenceProvider.s3-bundle-provider.
property.Bucket
Name
```

Value: my-bundles

- Name:

```
xml.providers.extensionBundlePersistenceProvider.s3-bundle-provider.
property.Credentials
Provider
```

Value: DEFAULT_CHAIN

- Name:

```
xml.providers.extensionBundlePersistenceProvider.s3-bundle-provider.
property.Access
Key
```

Value: *****

- Name:

```
xml.providers.extensionBundlePersistenceProvider.s3-bundle-provider.
property.Secret Access
```

Key
Value: *****

3. Save the changes.

After you finish

When you have finished migrating bundle storage configurations, proceed by reviewing NiFi components.

Related Information

[Migrating flow storage](#)

[Reviewing NiFi components](#)

Reviewing NiFi components

After you have completed the migration, it is a best practice to review the NiFi components (processors, controller services, and reporting tasks) for any cluster specific configurations so that you can update them before you start your data flows.

Procedure

1. Start NiFi service and access UI.



Tip:

In NiFi configuration, consider setting Flow Controller Auto Resume State (`nifi.flowcontroller.autoResumeState`) to false by unchecking it. This sets the state of all components to stopped on startup, allowing you to make any post migration flow adjustments before the components run.

2. Make the following changes as needed within the NiFi UI:

- If any components (processors, controller services, or reporting tasks) are configured specifically for the source cluster, edit for the destination cluster.
- Address any components (processors, controller services or reporting tasks) that are marked Invalid. For example:
 - `StandardSSLContextService` or `StandardRestrictedContextService` controller services may need to be updated to use the new certs setup for the destination cluster.
 - Any client controller services that connect to a server controller service will need to be updated. For example, `DistributedMapCacheClientService` and `DistributedMapCacheServer`.
 - You may need to update Kafka and Hive processor versions.
 - You may need to update HBase and HDFS processor configurations.

3. Start NiFi Registry and access UI.

Related Information

[Migrating bundle storage configurations](#)

Migrating file-based authorization to Ranger

Both NiFi and NiFi Registry services have the option to convert existing file-based provider policies to Ranger provider policies.

Migrating NiFi File-Based authorization to Ranger

You can convert existing file-based provider NiFi policies to Ranger provider policies.

Before you begin

The following steps assume that the Ranger service is installed in the CDP-DC cluster.

Procedure

1. Create any users and groups from the NiFi users.xml that do not already exist in Ranger.
2. Select Ranger as a dependency from NiFi configuration.
3. Restart NiFi.
4. Select Migrate File-based Authorizations to Ranger from the Actions drop-down. Confirm the action.
5. After a successful migration, verify that the policies are available in the NiFi Ranger service.

Migrating NiFi Registry file-based authorization to Ranger

You can convert existing file-based provider NiFi Registry policies to Ranger provider policies.

Before you begin

The following steps assume that the Ranger service is installed in the CDP-DC cluster.

Procedure

1. Create any users and groups from the NiFi Registry users.xml that do not already exist in Ranger.
2. Select Ranger as a dependency from NiFi Registry configuration.
3. Restart NiFi Registry.
4. Select Migrate File-based Authorizations to Ranger from the Actions drop-down. Confirm the action.
5. After a successful migration, verify that the policies are available in the NiFi Registry Ranger service.

In-place upgrade from HDF 3.5.x to CFM 2.x on CDP

This section describes the in-place upgrade process from HDF to CFM on CDP Private Cloud Base. Learn the general steps required to move your NiFi dataflow and NiFi Registry versioned flows from an HDF cluster to a CFM cluster on CDP Private Cloud Base.



Note: If you are using an HDF version lower than 3.5.2, you need to upgrade to HDF 3.5.2 first and then proceed with the in-place upgrade process to CFM. For details, see [Ambari Managed HDF Upgrade](#).

Before you upgrade

Before you start the HDF to CFM in-place upgrade process, ensure you understand the upgrade path available to you and your cluster meets the required prerequisites.

Upgrade path

Use the AM2CM tool to upgrade straight from HDF 3.5.2.0 to CFM 2.1.5 on CDP 7.1.7, CDP 7.1.7 SP1, CDP 7.1.7 SP2, or CDP 7.1.8.

For other upgrade scenarios, see the [Upgrade and migration paths](#).



Note: HDF and CFM are packaged with a different component list. For details, see [CFM component versions](#) and [HDF Component Support](#). Components that are part of HDF but not part of CFM will become part of the CDP package. For information on migrating workloads of HDF components that are not included in CFM into CDP, see [Migrating workloads](#) in the *CDP Private Cloud Upgrade Guide*.

Setting Ambari environment variables

Assign values to Ambari environment variables so that scripts used in the migration procedure can use values set for the variables.

About this task

You need to assign values to these environment variables because the scripts used in this migration use these predefined variables. If the values are not set, the scripts

Procedure

1. SSH to host where Ambari is installed.

The Ambari machine contains the Cloudera Manager server.

2. Assign values to the following variables:

```
export clustername=HDFTOCDF
export pfork=6
export ambariuser=admin
export ambaripwd=admin
export ambariport=8080
export backupdir=/data/backups
export ambariprotocol=http
export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.262.b10-0.el7_8.x86_64
export ambariserver=`hostname -f`
export metricscollectorhost=ccycloud-3.am2cmhdf.root.hwx.site
export grafanahost=ccycloud-3.am2cmhdf.root.hwx.site
export infrahost=ccycloud-1.am2cmhdf.root.hwx.site
export kdchost=`hostname -f`
export kdcrealm=EXAMPLE.COM
export kdcpasswd=Cloudera123
[ "${ambariprotocol}" = "https" ] && export securecurl="-k" || export securecurl=""
```

The following list explains the environment variables:

- export clustername=HDFTOCDF

This is the cluster name in the Ambari UI. An Ambari server could manage more than one cluster, and when that happens the cluster name identifies the cluster. This is important, because this value is used when calling Ambari REST API to export blueprint.

- export ambariuser=admin

Ambari user name which has right to perform Ambari REST calls and admin operations on the cluster.

- export ambaripwd=admin

Ambari password for the above user (the user name and password variables are used by curl bash scripts later in the documentation).

- export ambariport=8080

HTTP or HTTPs port number which is needed for Ambari REST API calls.

- export backupdir=/data/backups

The directory where you save the cluster blueprint.

- export ambariprotocol=http

Ambari server supported protocol (HTTP or HTTPS).

- export JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.262.b10-0.el7_8.x86_64

JAVA_HOME directory which is needed for some bash script.

- export ambariserver=`hostname -f`

Hostname of the machine where the Ambari server is running.

- export metricscollectorhost=ccycloud-3.am2cmhdf.root.hwx.site

Hostname of the metrics collector. Usually it is the machine where the Ambari server is running.

- `export grafanahost=ccycloud-3.am2cmhdf.root.hwx.site`
Hostname of the Grafana host. Usually it is the machine where the Ambari server is running.
- `export infrahost=ccycloud-1.am2cmhdf.root.hwx.site`
Hostname of the Infra host. Usually it is the machine where the Ambari server is running.
- `export kdchost=`hostname -f``
If Kerberos is used, this should point to the machine where Kerberos server is running.
- `export kdcrealm=EXAMPLE.COM`
If Kerberos is used, this should point to the Kerberos server realm.
- `export kdcpasswd=Cloudera123`
If Kerberos is used, this should point to Kerberos server admin password.
- `["${ambariprotocol}" = "https"] && export securecurl="-k" || export securecurl=""`
Initializes securecurl variable, based on previous values, used in some of the bash script later in the documentation.

Installing Solr service

Ranger uses Apache Solr service to store data. So, you must install Solr service on the cluster before the upgrade, if it is not installed already. You can do it through the Ambari UI.

Procedure

1. Go to **Services Add Service** on the Ambari UI.
The Choose Services window appears.
2. Select **Infra Solr** service in the wizard and click **Next**.
3. Follow the instructions to configure settings and advanced properties and install the service.
4. Click **Deploy**.

Checking cluster services

You must ensure that all services, you installed, are running in your cluster.

Perform one of the following tasks for checking if the cluster services are running:

- Manually from the Ambari UI

Run the service check for each service under **Service Actions Run Service Check**.

- Through Ambari API

```
curl ${securecurl} -u "${ambariuser}":"${ambaripwd}" -i -H 'X-Requested-By: ambari' -X PUT -d '{"RequestInfo": {"context": "Start All via REST"}, "Body": {"ServiceInfo": {"state": "STARTED"}}}' ${ambariprotocol}://${ambariserver}:${ambariport}/api/v1/clusters/${clustername}/services/
```

Restart all services. Check if time service stops and starts:

```
###STOP
# curl ${securecurl} -u "${ambariuser}":"${ambaripwd}" -i -H 'X-Requested-By: ambari' -X PUT -d '{"RequestInfo": {"context": "Stop All via REST"}, "Body": {"ServiceInfo": {"state": "INSTALLED"}}}' ${ambariprotocol}://${ambariserver}:${ambariport}/api/v1/clusters/${clustername}/services/
###START (takes a long time)
# curl ${securecurl} -u "${ambariuser}":"${ambaripwd}" -i -H 'X-Requested-By: ambari' -X PUT -d '{"RequestInfo": {"context": "Start All via REST"},
```

```
"Body": {"ServiceInfo": {"state": "STARTED"}}}' ${ambariprotocol}://${ambariserver}:${ambariport}/api/v1/clusters/${clustername}/services/
```

If any cluster service is not running, fix the service settings before you proceed.

Checking service accounts

You must ensure that all service accounts required for the upgrade are present. These service accounts include Ambari metrics user, smoke user, Hadoop group, infra solr user, NiFi user, NiFi Registry user, Ranger group, Ranger user, and ZooKeeper user.

Procedure

- Run the following command:
 - For service account: `cat /etc/passwd`
 - For groups: `cat /etc/group`
- Check if all the service accounts, mentioned in the following table, are present.

Name	Value
Ambari Metrics User	ams
Smoke User	ambari-qa
Hadoop Group	hadoop
Infra Solr User	infra-solr
Nifi User	nifi
Nifi Registry User	nifiregistry
Ranger Group	ranger (Optional. Needed when you use ranger service in Ambari)
Ranger User	ranger
ZooKeeper User	zookeeper

- If any service account is missing, create that account in your cluster.

Collecting data for upgrade

Collect passwords and Kafka broker IDs before you start the upgrade to make sure that required data is available during upgrade and post configuration.

Collecting Ranger passwords

If you use Ranger service, you need to collect the passwords for Ranger.

Collect the following passwords for Ranger:

- Ranger database password
The password used for ranger database
- Ranger admin password
The admin web interface password
- Ranger usersync password
- Ranger tagsync password
- Ranger keyadmin password
- Ranger usersync ldap password

If LDAP user sync setting is configured.

Collecting NiFi Registry database password

You need to migrate the NiFi Registry database password manually during the upgrade.

If you forget your database password, but you know your password which used to encrypt the NiFi Registry password, you can decrypt the `nifi.registry.properties` file properties with the following command:

```
/usr/hdf/current/nifi-toolkit/bin/encrypt-config.sh -r /usr/hdf/current/nifi-registry/conf/nifi-registry.properties --decrypt --nifiRegistry -v -p <<password_used_for_encryption>>
```

The output of the command contains the database password.

Extracting Kafka broker IDs

You must extract the Kafka broker IDs before you upgrade the HDF cluster. The Kafka broker IDs are used in upgrading HDF to CFM on CDP cluster, where you need to override the `kafka-broker-ids.ini` files with the created one.

You must extract the Kafka broker IDs manually from the HDF 3.5.2.0 cluster. When you upgrade to Cloudera Manager, you must manually enter the broker IDs to the real values in Kafka.

1. Create the `kafka-broker-ids.ini` file.
2. Navigate to each Kafka broker host > `$log.dirs/meta.properties` and collect the `broker.id` value.

Here, `$log.dirs` refers to an Ambari configuration value.

3. Copy hostname `broker.id` to the `kafka-broker-ids.ini` file.

An example of the file format is as follows:

```
ctr-e153-xxxxx-xxxx71.cloudera.site 1001
ctr-e153-xxxxx-xxxx72.cloudera.site 1002
ctr-e153-xxxxx-xxxx73.cloudera.site 1003
```

Related Information

[Kafka documentation](#)

Downloading and extending Ambari blueprint

You need to download and extend the Ambari blueprint which, after this step, will be converted into a Cloudera Manager Deployment template by the AM2CM tool.

About this task

Ambari blueprint is a JSON file that describes the entire cluster including number of clusters present, names of the clusters, services installed on those clusters, configuration details of the installed services, and so on. The JSON file helps to upgrade the cluster. The Ambari blueprint helps you to import or export your clusters if you want to replicate your clusters with the same configuration to any other cluster. You first need to download this JSON file from HDF, then extend the JSON file.

Downloading Ambari blueprint

Ambari blueprint is a JSON file that describes the entire cluster including number of clusters present, names of the clusters, services installed on those clusters, configuration details of the installed services, and so on. The JSON file helps to upgrade the cluster. The Ambari blueprint helps you to import or export your clusters if you want to replicate your clusters with the same configuration to any other cluster.

About this task

You need the Ambari configuration to install and configure Ranger. You need to get the Ambari blueprint.

Before you begin

If your cluster has Kerberos and SSL enabled, you need to disable them.

Procedure

1. Go to Dashboard Kerberos in the Ambari UI.
2. Run the following command to download the Ambari blueprint:

```
curl ${securecurl} -H "X-Requested-By: ambari" -X GET -u ${ambariuser}:${ambaripwd} ${ambariprotocol}://${ambariserver}:${ambariport}/api/v1/clusters/${clustername}?format=blueprint > "${backupdir}/${clustername}_blueprint_"$(date +%Y%m%d%H%M%S)".json
```

Extending the JSON file

After you download your Ambari blueprint, you have to extend the JSON file manually with the host details. You need to extend the JSON file in order to let Cloudera Manager know which component to install at which host.

Procedure

1. Check how many hostgroups exist in the blueprint.
2. Collect all machine hostnames and IP addresses.
3. Identify which machine belongs to which hostgroup.
4. Extend all hostgroups with the data which describe the belonging host with the following JSON format:

```
"hosts" : [
  {
    "hostname" : "machine host name",
    "hostipaddress" : "machine ip address"
  }
]
```



Note: The section should be properly inserted with comma to preserve the valid JSON format.

Installing and setting up Cloudera Manager

After you successfully complete the prerequisites, you can install and configure Cloudera Manager in your cluster.

Checking preinstallation setup

You must ensure that the Ambari blueprint, that you download, contains extended hostgroups with hosts, and hosts and all hostgroups contain host entries.

Procedure

1. Download the latest Ambari blueprint and extend hostgroups with hosts.
2. Ensure that the blueprint contains hosts included and all host groups contain host entries:

```
[root@ccycloud-1 backups]# egrep '"hostname" : ' /data/backups/HDFTOCDF_blueprint_with_hosts_20201103144235.json | sort
"hostname" : "ccycloud-1.am2cmhdf.root.hwx.site",
"hostname" : "ccycloud-2.am2cmhdf.root.hwx.site",
"hostname" : "ccycloud-3.am2cmhdf.root.hwx.site",
"hostname" : "ccycloud-4.am2cmhdf.root.hwx.site",
```

For more details,

```
egrep -C3 '"hostname" : ' /data/backups/HDFTOCDF_blueprint_with_hosts_20201103144235.json
```

```

    ],
    "hosts" : [
      {
        "hostname" : "ccycloud-2.am2cmhdf.root.hwx.site",
        "hostipaddress" : "172.27.133.196"
      }
    ],
  --
  ],
  "hosts" : [
    {
      "hostname" : "ccycloud-1.am2cmhdf.root.hwx.site",
      "hostipaddress" : "172.27.60.128"
    }
  ],
  --
  ],
  "hosts" : [
    {
      "hostname" : "ccycloud-4.am2cmhdf.root.hwx.site",
      "hostipaddress" : "172.27.92.133"
    }
  ],
  --
  ],
  "hosts" : [
    {
      "hostname" : "ccycloud-3.am2cmhdf.root.hwx.site",
      "hostipaddress" : "172.27.18.194"
    }
  ],
  ],

```

Configuring Cloudera Manager repository

You need to download the Cloudera Manager repository, configure the repository file, and copy the file on all hosts in the cluster. You configure the repository to deploy Cloudera Manager and daemons on the machines.

Procedure

1. Download the repository on the host currently running the Ambari server:

```
wget https://<username>:<password>@archive.cloudera.com/p/cm7/7.4.4/redhat7/yum/cloudera-manager.repo -P /etc/yum.repos.d/
```

2. Configure the repository file to include username and password for the Cloudera paywall:

```
# vi /etc/yum.repos.d/cloudera-manager.repo

[cloudera-manager]
name=Cloudera Manager 7.4.4
baseurl=https://archive.cloudera.com/p/cm7/7.4.4/redhat7/yum/
gpgkey=https://archive.cloudera.com/p/cm7/7.4.4/redhat7/yum/RPM-GPG-KEY-cloudera
username=changeme
password=changeme
gpgcheck=1
enabled=1
autorefresh=0
type=rpm-md
```

3. Copy the repository file on all hosts in the cluster:

```
for host in $(echo \
ccycloud-2.am2cmhdf.root.hwx.site \
ccycloud-3.am2cmhdf.root.hwx.site \
ccycloud-4.am2cmhdf.root.hwx.site \
); do scp /etc/yum.repos.d/cloudera-manager.repo $host:/etc/yum.repos.d/
cloudera-manager.repo ;done
```

Installing Cloudera Manager server and agents

You need to install Cloudera Manager server on the Ambari server host, and Cloudera Manager agents and daemons on all hosts. Otherwise, you will not be able to upgrade your cluster.

Procedure

1. Install Cloudera Manager server on the Ambari server host:

```
yum install cloudera-manager-server
```

2. Install Cloudera Manager agents and daemons on all hosts:

```
for host in $(echo \
ccycloud-1.am2cmhdf.root.hwx.site \
ccycloud-2.am2cmhdf.root.hwx.site \
ccycloud-3.am2cmhdf.root.hwx.site \
ccycloud-4.am2cmhdf.root.hwx.site \
); do ssh $host "yum install -y cloudera-manager-daemons cloudera-manager-
agent";done
```

Configuring database for Cloudera Manager

You need to configure the database for Cloudera Manager to use.

About this task

The examples you view in this section are for PostgreSQL. Cloudera Manager supports other databases as well. For more details about the syntax details of other databases, see *Syntax for scm_prepare_database.sh*.

Procedure

1. Configure the Reports Manager database:

```
[root@ccycloud-1 yum.repos.d]# sudo -u postgres psql
psql (9.2.24, server 10.12)
WARNING: psql version 9.2, server version 10.0.
        Some psql features might not work.
Type "help" for help.

postgres=# CREATE ROLE scm  LOGIN PASSWORD 'scm';
CREATE ROLE
postgres=# CREATE ROLE rman  LOGIN PASSWORD 'rman';
CREATE ROLE
postgres=# CREATE DATABASE rman OWNER rman ENCODING 'UTF8';
CREATE DATABASE
postgres=# CREATE DATABASE scm OWNER scm ENCODING 'UTF8';
CREATE DATABASE
postgres=#
```

2. Configure the Cloudera Manager database on the Cloudera Manager server host:

```
[root@ccycloud-1 yum.repos.d]# /opt/cloudera/cm/schema/scm_prepare_database.sh postgresql scm scm scm
JAVA_HOME=/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.262.b10-0.el7_8.x86_64
Verifying that we can write to /etc/cloudera-scm-server
Creating SCM configuration file in /etc/cloudera-scm-server
Executing: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.262.b10-0.el7_8.x86_64/bin/java -cp /usr/share/java/mysql-connector-java.jar:/usr/share/java/oracle-connector-java.jar:/usr/share/java/postgresql-connector-java.jar:/opt/cloudera/cm/schema/./lib/* com.cloudera.enterprise.dbutil.DbCommandExecutor /etc/cloudera-scm-server/db.properties com.cloudera.cmf.db.
log4j:ERROR Could not find value for key log4j.appender.A
log4j:ERROR Could not instantiate appender named "A".
[2020-11-05 11:17:18,802] INFO      0[main] - com.cloudera.enterprise.dbutil.DbCommandExecutor.testDbConnection(DbCommandExecutor.java) - Successfully connected to database.
All done, your SCM database is configured correctly!
[root@ccycloud-1 yum.repos.d]#
```

3. Check the db.properties file and ensure that the settings match with the settings mentioned in the following script:

```
[root@ccycloud-1 yum.repos.d]# cat /etc/cloudera-scm-server/db.properties
# Auto-generated by scm_prepare_database.sh on Thu Nov  5 11:17:18 PST 2020
#
# For information describing how to configure the Cloudera Manager Server
# to connect to databases, see the "Cloudera Manager Installation Guide."
#
com.cloudera.cmf.db.type=postgresql
com.cloudera.cmf.db.host=localhost
com.cloudera.cmf.db.name=scm
com.cloudera.cmf.db.user=scm
com.cloudera.cmf.db.setupType=EXTERNAL
com.cloudera.cmf.db.password=scm
[root@ccycloud-1 yum.repos.d]#
```

Related Information

[Syntax for scm_prepare_database.sh](#)

Starting Cloudera Manager server and adding license

To use Cloudera Manager after the installation, you must start the Cloudera Manager server, open the Cloudera Manager web UI, and add Cloudera Manager license.

Procedure

1. Start Cloudera Manager server:

```
[root@ccycloud-1 yum.repos.d]# systemctl start cloudera-scm-server
[root@ccycloud-1 yum.repos.d]# systemctl enable cloudera-scm-server
[root@ccycloud-1 yum.repos.d]# tail -f /var/log/cloudera-scm-server/cloud-era-scm-server.log

####Look for  `Started Jetty Server`
```

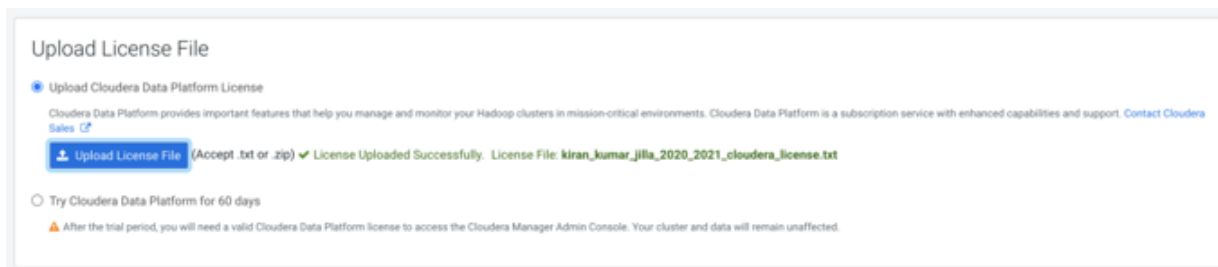
2. Open the Cloudera Manager web UI on browser:

<http://ccycloud-1.am2cmhdf.root.hwx.site:7180/cmf/login>

Username: admin

Password: admin

3. Add the Cloudera Manager license.



4. Click Cloudera Manager on the top-left corner of your screen to exit the adding cluster step.

Do not click Add Cluster using Wizard. If you click, it starts a wizard where you can create a cluster through the UI and not through the blueprint which is created by the AM2CM tool.

Configuring Cloudera agents and hosts

You need to add Cloudera agents and hosts before you perform the HDF upgrade.

About this task

You can either automate the process of configuring Cloudera agents and hosts, or manually perform the configuration.

- To automate, change the agents configuration file before adding hosts to Cloudera Manager:

```
for host in $(echo \
ccycloud-1.am2cmhdf.root.hwx.site \
ccycloud-2.am2cmhdf.root.hwx.site \
ccycloud-3.am2cmhdf.root.hwx.site \
ccycloud-4.am2cmhdf.root.hwx.site \
); do ssh $host "sed -i "s/server_host=localhost/server_host=ccycloud-1
.am2cmhdf.root.hwx.site/" /etc/cloudera-scm-agent/config.ini
";done

for host in $(echo \
ccycloud-1.am2cmhdf.root.hwx.site \
ccycloud-2.am2cmhdf.root.hwx.site \
ccycloud-3.am2cmhdf.root.hwx.site \
ccycloud-4.am2cmhdf.root.hwx.site \
); do ssh $host "systemctl restart cloudera-scm-agent";done

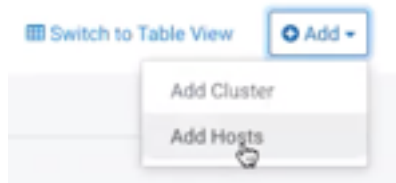
for host in $(echo \
ccycloud-1.am2cmhdf.root.hwx.site \
ccycloud-2.am2cmhdf.root.hwx.site \
ccycloud-3.am2cmhdf.root.hwx.site \
ccycloud-4.am2cmhdf.root.hwx.site \
); do ssh $host "systemctl status cloudera-scm-agent";done
```

- To manually configure agents and hosts, perform the following steps:

Procedure

1. Go to the Cloudera Manager UI.

2. Click **Add Add Hosts** at the top-right corner of your screen.



The Add Hosts window appears.

Add Hosts

The Add Hosts Wizard allows you to install the Cloudera Manager Agent on new hosts. You can either keep the new hosts available to be added to a cluster in the future, or you can add new hosts to an existing cluster

☒ **Add hosts to Cloudera Manager**
You can use these hosts later to create new clusters or expand existing clusters.

☐ **Add hosts to Cluster**
Cluster 1

3. Select **Add hosts to Cloudera Manager** and click **Continue**.

Do not add hosts to the cluster yet.

The Setup Auto-TLS screen appears.

4. Click **Continue**.

The Specify Hosts screen appears.

Add Hosts to Cloudera Manager

Specify Hosts

Specify hosts for your cluster installation. Hosts should be specified using the same hostname (FQDN) that they will identify themselves with.

Hostname

Hint: Search for hostnames or IP addresses using patterns

SSH Port:

5. Specify the hostname and click **Search**.

The hostname appears.

6. Select the host and click **Continue**.

The Select Repository screen appears.

Add Hosts to Cloudera Manager

Select Repository

Cloudera Manager Agent

Cloudera Manager Agent 7.3.1 (#10891891) needs to be installed on all new hosts.

Repository Location ☒ **Public Cloudera Repository**
Ensure the above version is listed in <https://archive.cloudera.com/cdm7> and that you have access to that repository. Requires direct Internet access on all hosts.

☐ Custom Repository

7. Select Public Cloudera Repository and click Continue.

The Select JDK screen appears.

Add Hosts to Cloudera Manager

- Setup Auto-TLS
- Specify Hosts
- Select Repository
- Select JDK**
- Enter Login Credentials
- Install Agents
- Inspect Hosts for Correctness

Select JDK

Major Version	Cloudera Runtime 7	CDH 6	CDH 5
Minor Version	7.1 and above	7.0 and above	6.3 and above
Supported JDK Version	OpenJDK 8, 11 or Oracle JDK 8, 11	OpenJDK 8 or Oracle JDK 8	OpenJDK 8 or Oracle JDK 8

If you plan to use JDK 11, you will need to install it manually on all hosts and then select the **Manually manage JDK** option below.

☐ Manually manage JDK

☒ Please ensure that a supported JDK is **already installed** on all hosts. You will need to manage installing the unlimited strength JCE policy file, if necessary.

☐ Install a Cloudera-provided version of OpenJDK

By proceeding, Cloudera will install a supported version of OpenJDK version 8.

☐ Install a system-provided version of OpenJDK

By proceeding, Cloudera will install the default version of OpenJDK version 8 provided by the Operating System.

[More details on supported JDK version.](#)

8. Select Manually manage JDK and click Continue.

The Enter Login Credentials screen appears.

Add Hosts to Cloudera Manager

- Setup Auto-TLS
- Specify Hosts
- Select Repository
- Select JDK
- Enter Login Credentials**
- Install Agents
- Inspect Hosts for Correctness

Enter Login Credentials

Root access to your hosts is required to install the Cloudera packages. This installer will connect to your hosts via SSH and log in either directly as root or as another user with password-less sudo/brun privileges to become root.

Login To All Hosts As: ☒ root ☐ Another user

You may connect via password or public-key authentication for the user selected above.

Authentication Method: ☒ All hosts accept same password ☐ All hosts accept same private key

Enter Password:

Confirm Password:

SSH Port:

Number of Simultaneous Installations:

(Running a large number of installations at once can consume large amounts of network bandwidth and other system resources)

9. Enter your login credentials and click Continue.

The Install Agents screen appears where the progress of the installation process is displayed.

Add Hosts to Cloudera Manager

- Setup Auto-TLS
- Specify Hosts
- Select Repository
- Select JDK
- Enter Login Credentials
- Install Agents**
- Inspect Hosts for Correctness

Install Agents

Installation in progress.

0 of 1 host(s) completed successfully. [Abort Installation](#)

Hostname	IP Address	Progress	Status
ctr-e168-1619641223258-12905-01-000007.hwx.site	172.27.161.6	<div></div>	Installing cloudera-manager-agent package...

After the installation process is complete, the Inspect Hosts for Correctness screen appears. In this step, Cloudera Manager inspects the settings of your host and displays some suggestions.

10. Validate your settings and click Finish.

Adding and configuring Cloudera Management Services

You need to add and configure Cloudera Management Services before you perform the HDF upgrade.

Adding Cloudera Management Services

Learn how to add Cloudera Management Services.

Procedure

1. Log in to the Cloudera Manager UI.

2. Click Add Add Cloudera Management Service .



Note: If you run into any issues at this step while you are upgrading to Cloudera Manager, you need to add Cloudera Management services manually, using a REST API.

- a. Use a REST API client like Postman or curl to call the following URL with PUT method:

`http://CM_SERVER_IP:7180/api/v49/cm/service`

With raw/JSON: {}

(empty object)

Headers:

Content-Type: application/json

Curl example:

```
curl -X PUT -d '{}' -H "Content-Type: application/json" http://CM_SERVER_URL_OR_IP:7180/api/v49/cm/service --user CM_USER:CM_PASSWORD
```

The Cloudera Management Service appears on the Cloudera Manager UI.

- b. Select Cloudera Management Service and click Actions Add Role Instances .

The Assign Roles screen appears.

Assign Roles

You can customize the role assignments for your new service here, but note that if assignments are made incorrectly, such as assigning too many roles to a single host, performance will suffer.

You can also view the role assignments by host. [View By Host](#)

Service Monitor × 1 New ccycloud-1.am2cmhdf.root.hwx.site	Activity Monitor Select a host	Host Monitor × 1 New ccycloud-1.am2cmhdf.root.hwx.site
Reports Manager × 1 New ccycloud-1.am2cmhdf.root.hwx.site	Event Server × 1 New ccycloud-1.am2cmhdf.root.hwx.site	Alert Publisher × 1 New ccycloud-1.am2cmhdf.root.hwx.site
Navigator Audit Server Select a host	Navigator Metadata Server Select a host	Telemetry Publisher Select a host

3. Customize the role assignments and click Continue.

The Setup Database screen appears.

4. Configure and test the database connections, and click Continue.

The Review Changes screen appears.



Note: The database connection configuration is needed only for Cloudera Manager versions older than 7.4.4. Starting from Cloudera Manager version 7.4.4, you do not have to configure database connection.

5. Review the changes and click Continue.

The Command Details screen appears.

6. Click Continue after the commands run successfully.

The Summary screen appears.

7. Check the summary and click Finish.

Modifying host monitor port number

If you use NiFi with security enabled, then the Cloudera host monitor port number and the NiFi port number might be the same. That causes problems. So before continuing, you must modify it and then restart the Cloudera Management service.

Procedure

1. Go to the Cloudera Manager main dashboard.
2. Select Cloudera Management Service.
3. Click Configuration.
4. Search for the Host Monitor Web UI HTTPS Port configuration value.

5. If the value is 9091, modify it to 9092 or other free port numbers, and save the configuration.
6. Restart Cloudera Management Service.

Upgrading HDF to CFM on CDP Private Cloud Base

Cloudera provides an AM2CM tool to enable you to upgrade HDF cluster to CFM cluster on CDP Private Cloud Base easily. You need to download the AM2CM tool, configure settings for the tool, and generate the Cloudera Manager Deployment template. After you generate the Cloudera Manager Deployment template, you need to configure the parcel, deploy Cloudera Manager, add CFM parcel, and activate the parcel through the Cloudera Manager UI.

Generating Cloudera Manager Deployment template

Learn how to generate the Cloudera Manager Deployment template that will then be used to upgrade your cluster.

Procedure

1. Ensure that you have the latest Ambari blueprint (JSON file).
2. Download the AM2CM tool:

```
wget https://archive.cloudera.com/am2cm/2.3.0.0/redhat7/yum/tars/am2cm/am2cm-tool-2.3.0.0-60.tar.gz
```

3. Extract the TAR file:

```
tar -xvf am2cm-tool-2.3.0.0-60.tar.gz
```

4. Check the project files:

```
# cd am2cm-2.3.0.0-60
# [root@ccycloud-1 am2cm-2.3.0.0-60]# ls
am2cm-2.2.0.2.3.0.0-60.jar  am2cm.sh  conf  lib  ranger_policy_migration.py
y

# [root@ccycloud-1 am2cm-2.3.0.0-60]# ls conf
ambari_blueprint_hostgroups.json  blueprint.json  cm-config-mapping-custom
m.ini  cm-config-mapping.ini  log4j.properties  ranger_policy_migration.py
service-config.ini  service-default-config.ini  user-settings.ini

# mv conf/blueprint.json conf/blueprint_initial.json
# [root@ccycloud-1 am2cm-2.3.0.0-60]# ls conf
ambari_blueprint_hostgroups.json  blueprint_initial.json  cm-config-mappi
ng-custom.ini  cm-config-mapping.ini  log4j.properties  ranger_policy_mi
gration.py  service-config.ini  service-default-config.ini  user-setting
s.ini
```

5. Set JAVA_HOME if not already done:

```
[root@ccycloud-1 am2cm-2.3.0.0-60]# echo $JAVA_HOME
/usr/lib/jvm/java-1.8.0-openjdk-1.8.0.262.b10-0.el7_8.x86_64
```

6. Copy your Ambari blueprint to the right path, that is the conf directory of the AM2CM tool, from the Home directory:

```
cp /data/backups/HDFTOCDF_blueprint_with_hosts_20201103144235.json /
root/am2cm-2.3.0.0-60/conf
```

7. Configure the user-settings.ini file:

It is really important to specify the cluster.name and cluster.displayname to the same original Ambari cluster name. Also, it is worth to fill all password field.

```
# Cluster details
cluster.name=zt-HDFTOCD
cluster.displayname=HDFTOCD
# Update this with correct CM/CDP version before running the tool
cluster.fullversion=7.1.7

cluster.originating_source=AM2CM version 2.0

# This is needed to update Ranger/Atlas policy service names.
ambari.cluster.name=HDFTOCD

#This flag enables to translate Config Groups to Role config groups.
cm.rolegroups.enable = false

# FROM JDBC URL - tool reads only Hostname and port number - rest of the
details like Database type, Database name, database user reads from from
Blueprint.
# Do not enclose double quotes for passwords, provide exact password.

# Hive JDBC settings
SERVICE_HIVE_hive_metastore_database_password=password
SERVICE_HIVE_hive_jdbc_url_override=JDBC_URL

# Oozie JDBC settings
SERVICE_OOZIE_oozie_database_password=password
SERVICE_OOZIE_oozie_service_JPAService_jdbc_url=JDBC_URL
# Ranger JDBC settings
SERVICE_RANGER_ranger_database_password=password
SERVICE_RANGER_rangeradmin_user_password=password
SERVICE_RANGER_rangerusersync_user_password=password
SERVICE_RANGER_rangertagsync_user_password=password
SERVICE_RANGER_rangerkeyadmin_user_password=password
SERVICE_RANGER_ranger_usersync_ldap_ldapbindpassword=password

SERVICE_RANGER_KMS_ranger_kms_master_key_password=password
SERVICE_RANGER_KMS_ranger_kms_database_password=password
#Knox Settings
SERVICE_KNOX_gateway_master_secret=admin
```

8. If your cluster contains Kafka service, please override the kafka-broker-ids.ini files with the one you created.

For more details, see [Extracting Kafka broker IDs](#).

9. Run the command to generate the Cloudera Manager template:

```
# cd am2cm-2.3.0.0-60
# chmod +x ./am2cm.sh
[root@ccycloud-1 am2cm-2.3.0.0-60]# ./am2cm.sh -bp conf/HDFTOCD_blueprint_with_hosts_20201103144235.json -dt conf/cm_deployment_template.json --source-version=HDF352

INPUT Ambari Blueprint : conf/HDFTOCD_blueprint_with_hosts_20201103144235.json
OUTPUT CM Template      : conf/cm_deployment_template.json

Starting blueprint to CM Template migration
Total number of hosts in blueprint: 4
```

```

Your cluster has services (listed below) that are not handled by this mi
gration tool.
AMBARI_METRICS
The tool will skip the above identified service related configs.

** This migration tool is a technical preview only **

Do you want to proceed with migration (Y OR N)? (N):
Y
Processing: LIVY
Processing: SOLR
Processing: TEZ
Processing: HDFS
Processing: OOZIE
Processing: SQOOP_CLIENT
Processing: NIFIREGISTRY
Processing: ZOOKEEPER
Processing: HBASE
Processing: YARN
Processing: RANGER_KMS
Processing: KNOX
Processing: ATLAS
Processing: HIVE_ON_TEZ
Processing: RANGER
Processing: HIVE
Processing: KAFKA
Processing: NIFI
Processing: SPARK_ON_YARN
Adding: QUEUEMANAGER
CM Template is generated at : /root/am2cm-2.3.0.0-60/conf/cm_deployment_
template.json
Successfully completed

```

10. Check for errors in the tool log for Cloudera Manager migration:

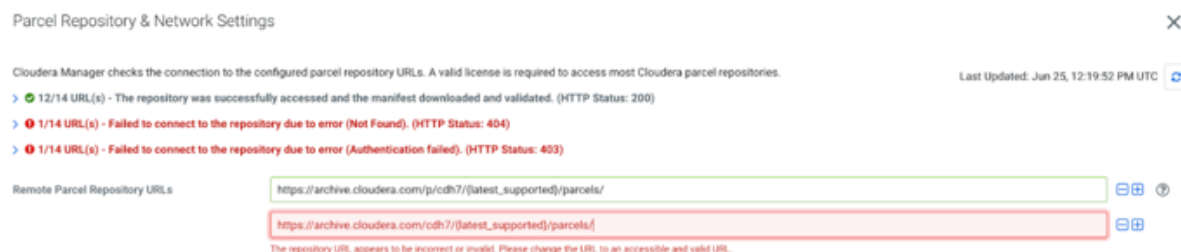
```
less am2cm-2.3.0.0-60/cm_migration.log
```

Configuring parcels

You need to check parcel repositories and network settings, and remove parcels that you do not need.

Procedure

1. Click Parcels in the left navigation pane of the Cloudera Manager UI.
2. Click Parcel Repositories and Network Settings and check for parcels with errors.



3. Remove parcels that you do not need.



4. Click Save and Verify Configuration.

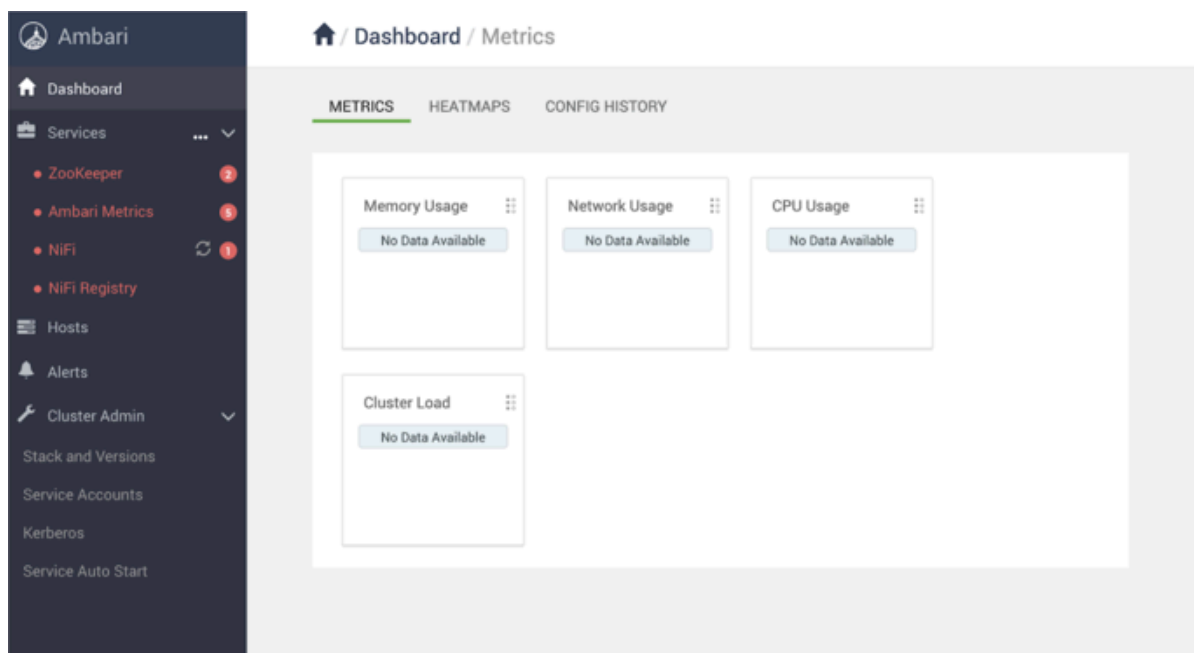
Deploying Cloudera Manager

You need to deploy Cloudera Manager for using it. To deploy Cloudera Manager, you need to stop all HDP services, download the CFM custom service descriptor files, deploy the existing cluster on CDP, and refresh Cloudera Manager.

Procedure

1. Stop all HDP services from Ambari.

The following image shows that all HDF services are stopped.



2. Download the CFM Custom Service Descriptor files:

```
cd /opt/cloudera/csd
wget https://<username>:<password>@archive.cloudera.com/p/cfm2/2.1.5/red
hat7/yum/tars/parcel/NIFI-<version>-<build>.jar
wget https://<username>:<password>@archive.cloudera.com/p/cfm2/2.1.5/red
hat7/yum/tars/parcel/NIFIREGISTRY-<version>-<build>.jar
chown cloudera-scm:cloudera-scm ./*
chmod 644 ./*
systemctl restart cloudera-scm-server
```

3. Deploy the existing cluster on CDP, using the Cloudera Manager Deployment template:

```
# cd am2cm-2.3.0.0-60/conf
[root@ccycloud-1 conf]# curl --user admin:admin -k -X PUT -H "Content-Type: application/json" -d @cm_deployment_template.json 'http://ccycloud-1.am2cmhdf.root.hwx.site:7180/api/v41/cm/deployment?deleteCurrentDeployment=false'
{
  "message" : "\"Role name 'nifiregistry-NIFI_REGISTRY_SERVER-108d5ac5472829ddbbaa38dfb3fd8ad' is not compliant. Use 'nifi0cb063fc-NIFI_REGISTRY_SERVER-108d5ac5472829ddbbaa38dfb3fd8ad', or do not use a name of the format <service name>-<roletype>-<arbitrary value>.\""}
[root@ccycloud-1 conf]# vi cm_deployment_template.json

Update nifiregistry-NIFI_REGISTRY_SERVER-108d5ac5472829ddbbaa38dfb3fd8ad to nifiregistry-NIFI_REGISTRY_SERVER

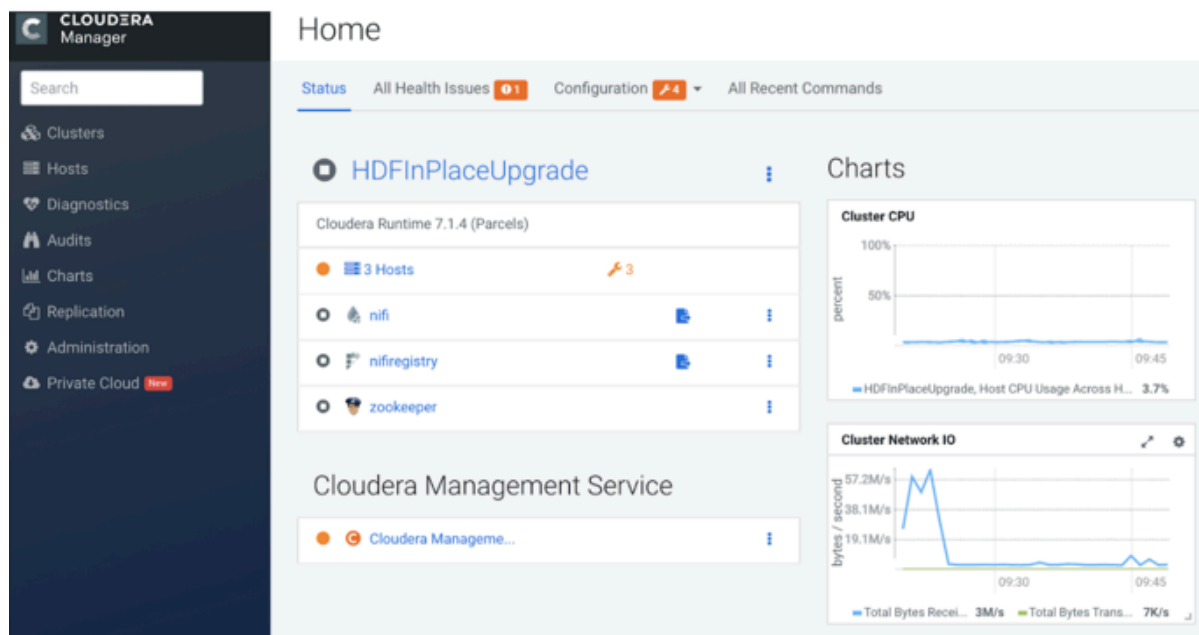
# curl --user admin:admin -k -X PUT -H "Content-Type: application/json" -d @cm_deployment_template.json 'http://ccycloud-1.am2cmhdf.root.hwx.site:7180/api/v41/cm/deployment?deleteCurrentDeployment=false'
```



Note: You must not forget to edit the `cm_deployment_template.json` file and rename `nifiregistry-NIFI_REGISTRY_SERVER-*` to `nifiregistry-NIFI_REGISTRY_SERVER`.

4. Refresh the Cloudera Manager browser.

The following image shows that the cluster is running with components including NiFi, NiFi Registry, and Zookeeper.



Adding CFM parcel in Cloudera Manager

You need to add the CFM parcel after you deploy Cloudera Manager. To add the CFM parcel, you need to download the CSD files for the CFM parcel and add an additional parcel URL for CFM in Cloudera Manager.

Procedure

1. Go to the Cloudera Manager UI and click Parcels in the left navigation pane.
2. Click Parcel Repositories and Network Settings.

3. Extend the Remote Parcel Repository URLs part, with the following additional parcel URL:

```
https://archive.cloudera.com/p/cfm2/2.1.5/redhat7/yum/tars/parcel/
```

4. Click Save and Verify Configuration.

The parcel URL depends on the operating system. To choose the appropriate URL, see *Download locations*.

Related Information

[Download locations](#)

Activating parcel

After you add the CFM parcel, you need to activate both the CFM and the Cloudera Runtime parcels.

Procedure

1. Click Parcels in the left navigation pane of the Cloudera Manager UI.
2. Find out the parcel corresponding to the required version of Cloudera Runtime and CFM.

The screenshot shows the Cloudera Manager UI for Cluster 1. The 'Parcels' section is active, showing a list of parcels. The left sidebar has filters for 'Location' (Cluster 1, Available Remotely) and 'Status' (Distributed, 2). The main table lists the following parcels:

Parcel Name	Version	Status	Action
ACCUMULO	1.9.2-1.ACCUMULO6.1.0.p0.908695	Available Remotely	Download
Cloudera Runtime	7.1.6-1.cdh7.1.6.p0.10506313	Distributed, Activated	Deactivate
CDH 6	6.3.4-1.cdh6.3.4.p0.6626826	Available Remotely	Download
CDH 5	5.16.2-1.cdh5.16.2.p0.8	Available Remotely	Download
CFM	2.1.1.0-13	Distributed, Activated	Deactivate
KAFKA	4.1.0-1.4.1.0.p0.4	Available Remotely	Download
KEYTRUSTEE_SERVER	7.1.6.0-1.keytrustee7.1.6.0.p0.10506313	Available Remotely	Download
KUDU	1.4.0-1.cdh5.12.2.p0.8	Available Remotely	Download
SQOOP_NETEZZA_CONNECTOR	1.5.1c6	Available Remotely	Download
	1.5.1c5	Available Remotely	Download

3. Click Download, Distribute, and Activate.

Only the valid and available button appears for a parcel and the order is download, distribute, and then activate.

Troubleshooting HDF upgrade

Learn about issues that might occur during the upgrade process, and how to resolve them.

Issue: Host health error

The host health shows the following error:

```
Clock Offset Suppress...The host's NTP service could not be located or did not respond to a request for the clock offset.
```

Solution:

Enable and start the NTP service for all hosts:

```
yum install -y ntp
```



```
systemctl start ntpd.service
systemctl enable ntpd.service
systemctl status ntpd.service
```

Otherwise perform the following steps:

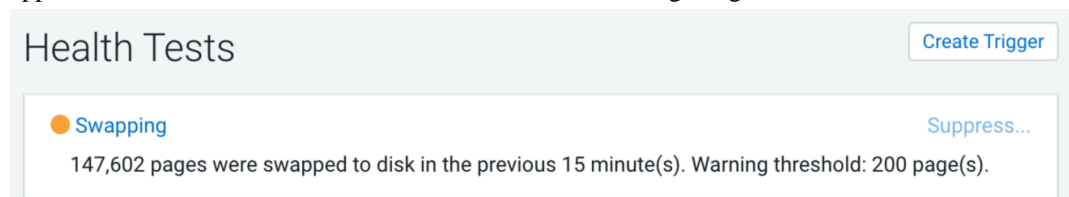
1. Click on the hosts health error, and navigate to the Host Clock Offset Threshold configuration.
2. Configure the values for thresholds to Never.



3. Click Save Changes.

Issue: Swappiness

The host check process checks how many pages were swapped in a given time, and an error might appear if that number is not reached, as shown in the following image:



Solution:

Set swappiness to one for all hosts.

```
cat /proc/sys/vm/swappiness
echo 1 > /proc/sys/vm/swappiness
cat /proc/sys/vm/swappiness
```

It suppresses the swappiness alerts for the cluster. Additionally, you can perform the following checks:

- Check configuration warnings for each service.
- Review JVM parameters and configuration for all services (some services are not transitioned).
- Review the Log4J configuration such as logs dir, size, and backup index.

Issue: Cloudera agent security

You might experience an issue with Cloudera agent security and Cloudera agents might not connect to the server.

Solution:

You can perform the following checks:

- Fix ownership of /var/lib/cloudera-scm-agent/agent-cert
- chown cloudera-scm:cloudera-scm /var/lib/cloudera-scm-agent/agent-cert
- chmod 755 /var/lib/cloudera-scm-agent/agent-cert

Post-upgrade steps on CDP

After you complete the upgrade of HDF, you need to verify or modify properties for NiFi Registry, NiFi, Ranger, Solr, and Zookeeper, migrate Kafka Ranger policies, and enable security if you want to use a secure cluster. You also need to set core configuration service, and configure Yarn.

Enabling security

If your previous cluster was secure or if you want to use a secure cluster, you have to enable Kerberos and auto-TLS.

For information about how to enable Kerberos and auto-TLS, see *Encrypting Data in Transit* and *Security Kerberos Authentication Overview*.

After setting up Kerberos, go to **Administration Security** and click **Generate missing credentials** on the **Kerberos Credentials** tab. You might get the following error depending on your Kerberos server settings:

Generate Missing Credentials

Status **Failed** Feb 28, 3:05:24 PM 1.54s

Encountered error with /opt/cloudera/cm/bin/gen_credentials.sh: Cannot access generated keytab file /var/run/cloudera-scm-server/cm2666535775818569378.keytab

If you get this error, it means that the principals generated by Ambari have a different maximum renewable ticket time what Cloudera Manager wants to use, which causes this error. To fix this you have to modify the principals created by Ambari to have the same maximum renewable ticket time what Cloudera Manager wants to use (5 days):

```
# Get a keytab where the user have right to modify principals
# kadmin -q "ktadd -k /tmp/admin.keytab -norandkey admin/admin@HDF.COM" -p
admin/admin@HDF.COM
# Get principals generated by ambari via ambari rest api call
principals=$(curl -H "Content-Type: text/csv" "${ambariprotocol}://${ambariuser}:${ambaripwd}@${ambariserver}:${ambariport}/api/v1/clusters/${clustername}/kerberos_identities?format=csv" | tail -n +2 | awk -F , '{ print $3}'
)
# Modify principal maxrenewlife to 5 day
for princ in "${principals[@]}"
do
    kadmin -k -t /tmp/admin.keytab -p admin/admin@HDF.COM -q "modprinc -ma
xrenewlife 432000 $princ"
done
#Delete keytab for security reasons
#rm -f /tmp/admin.keytab
```

Related Information

[Encrypting Data in Transit](#)

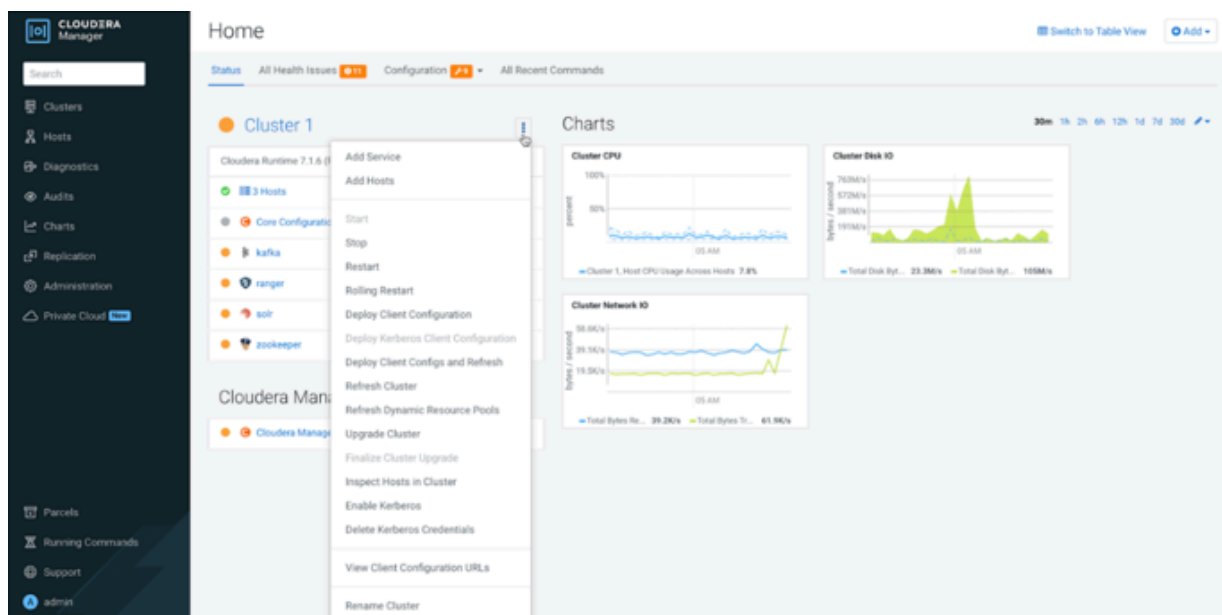
[Security Kerberos Authentication Overview](#)

Setting core configuration service

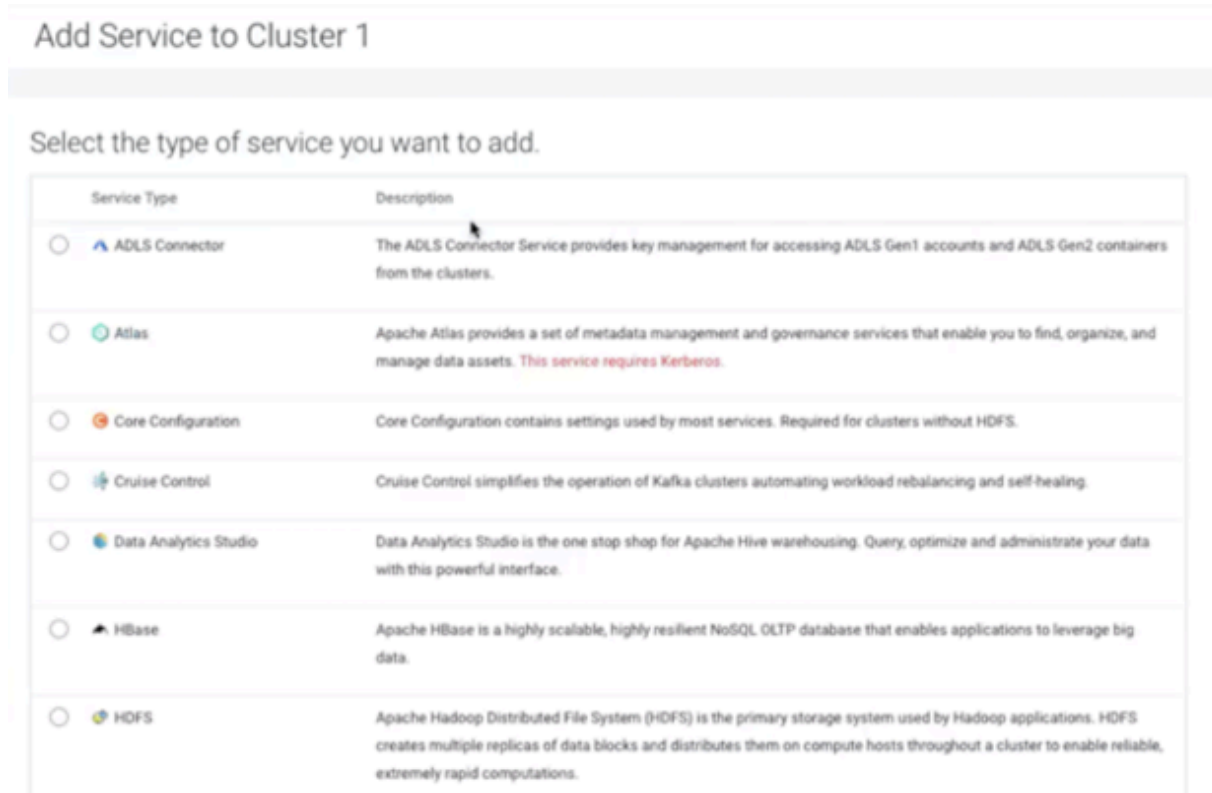
You need to add core configuration service to your cluster manually. The core configuration service allows you to create clusters without the HDFS service.

Procedure

1. Click Add Service in the Cloudera Manager UI.



The Add Service to Cluster window appears.



2. Select Core Configuration and click Continue.
3. Verify the assigned roles and click Continue.
4. Review the changes and click Continue.

The commands run to add core configuration settings.

5. Click Continue after all commands execute successfully.

6. Check the summary and click Finish.

Starting Zookeeper service

You must configure the Zookeeper server hosts to start the Zookeeper service.

Procedure

1. Add `zookeeper.snapshot.trust.empty=true` to your server configuration file.

This can be set in the `zoo.cfg` advanced configuration snippet in the Cloudera Manager UI.

2. Start the server.

Cloudera recommends removing the `zookeeper.snapshot.trust.empty` property after you have a working server.

3. Remove or move the `myid` file from the Zookeeper server hosts.

The path is `${dataDir}/myid`. For example, `# mv /hadoop/zookeeper/myid hadoop/zookeeper/myid.bak`.

It is possible that the `myid` file is in a different path. To determine the exact path, check Zookeeper service data directory configuration value in the Cloudera Manager UI.

4. Start the Zookeeper service from Cloudera Manager.

Configuring NiFi Registry settings

Before you first start NiFi Registry, you need to set the database password for NiFi Registry in Cloudera Manager and you need to migrate the NiFi Registry directories. If you have Kerberos enabled and Ranger installed, you also need to configure those for NiFi Registry before the first start.

Setting database password for NiFi Registry

Before the first starting of NiFi Registry, you need to set the NiFi Registry database password in Cloudera Manager. This password was collected from the Ambari-managed cluster earlier in the in-place upgrade process.

Before you begin

You have collected the NiFi Registry database password.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select NiFi Registry.
3. Go to the Configuration tab.
4. Search for the NiFi Registry Database Password configuration and specify the password that you have earlier collected.

Configuring Kerberos for NiFi Registry

After you enable Kerberos, as described in *Enable security*, you have to enable Kerberos for NiFi Registry and configure the initial admin identity setting, if the initial admin identity setting was configured before migration.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select NiFiRegistry.
3. Go to the Configuration tab.
4. Search for the Enable Kerberos Authentication configuration value and enable it.
5. Optional. Search for the Initial Admin Identity configuration and specify the correct principal name.

Configuring Ranger for NiFi Registry

If your cluster contains Ranger, then you need to configure Ranger service, before starting NiFi Registry.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select NiFiRegistry.
3. Go to the Configuration tab.
4. Search for the RANGER Service configuration and enable it.
5. Modify the ranger.plugin.nifi-registry.service.name property to match with the new ranger service name.

Modifying the service name in Ranger

If your NiFi Registry service name in Ranger contains the - character, then you must change it to the _ character.

Procedure

1. Go to Ranger Admin Web UI.
2. Go to resource based policies.
3. Click the edit button and modify the necessary characters in the service name and display name.
4. Click Save.

Migrating NiFi Registry directories

You must create a working directory to start the NiFi Registry service. The directory is the place where NiFi Registry stores existing buckets, configuration files, database files, and so on.

Procedure

1. Create a working directory for NiFi Registry and copy the content from the old directory:

```
mkdir /var/lib/nifiregistry
cp -R /var/lib/nifi-registry/* /var/lib/nifiregistry
chmod 755 /var/lib/nifiregistry
chown -R nifiregistry:nifiregistry /var/lib/nifiregistry
```

2. Start the NiFi Registry service.

Verifying Ranger configurations

You need to verify Ranger configurations before you start using the Ranger service.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select Ranger.
3. Click Configuration.
4. Ensure that the following configurations have actual database host, user and password used by Ranger:
 - Ranger Database Host (ranger_database_host)
 - Ranger Database User (ranger_database_user)
 - Ranger Database User Password (ranger_database_password)
5. Ensure that the load_balancer_url configuration point to the proper hostname and contains the correct schema and port number.

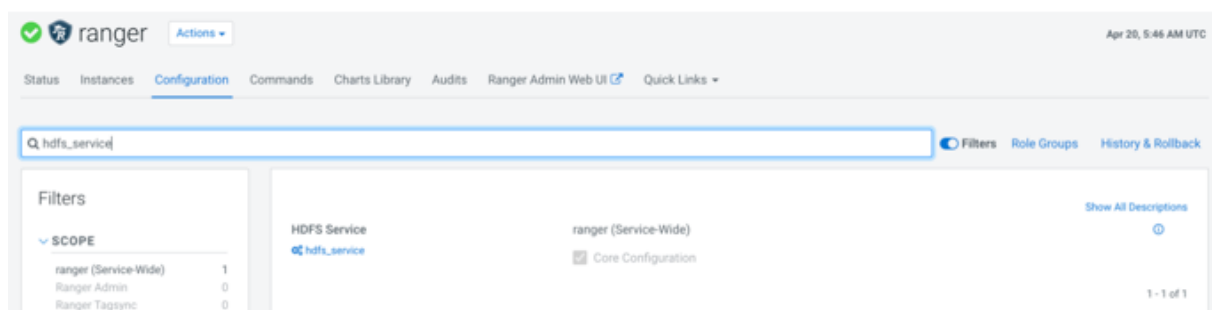
Otherwise, services will not be able to communicate with Ranger.

Configuring Ranger settings

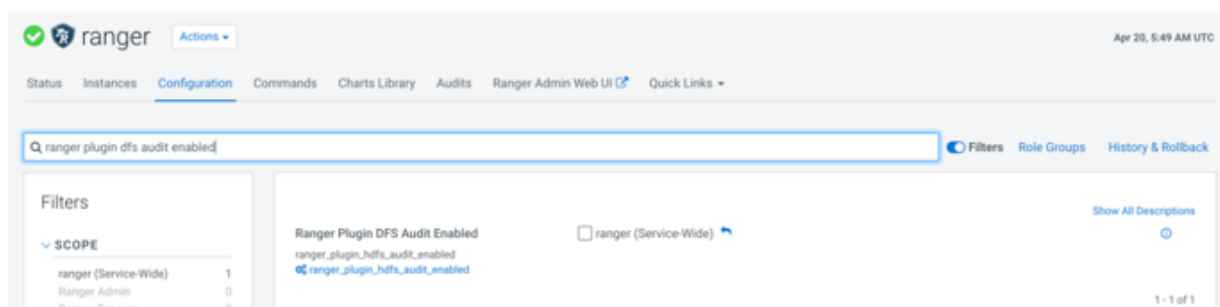
You need to configure Ranger settings and execute Ranger actions.

Procedure

1. Click ranger in the Cloudera Manager UI.
2. Go to the Configuration tab.
3. Locate the HDFS Service (hdfs_service) property and select the Core Configuration option.

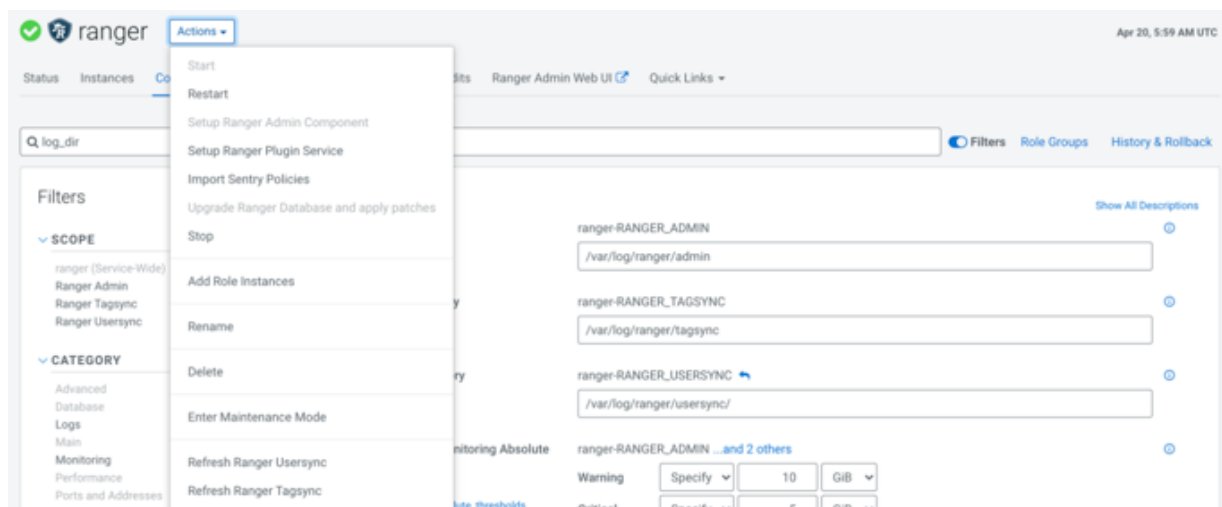


4. Click Save Changes.
5. Locate the Ranger Plugin HDFS Audit Enabled (ranger_plugin_hdfs_audit_enabled) property and set to false.



6. Locate and configure the following properties:
 - Ranger Admin User Initial Password
rangeradmin_user_password=<yourpassword>
 - Ranger Usersync User Initial Password
rangerusersync_user_password=<yourpassword>
 - Ranger Tagsync User Initial Password
rangertagsync_user_password=<yourpassword>
 - Ranger KMS Keyadmin User Initial Password
keyadmin_user_password=<yourpassword>
 - Ranger Database User Password
ranger_database_password=<yourpassword>
 - Ranger Usersync Log Directory
log_dir=/var/log/ranger/usersync/
7. Click Actions Upgrade Ranger Database and apply patches .

8. Click Actions Setup Ranger Admin Component .

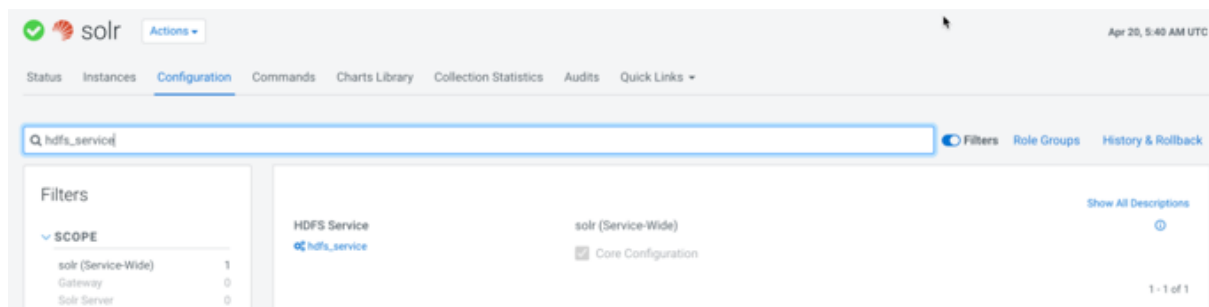


Configuring Solr settings

You need to configure Solr settings before you start using the Solr service. You also need to start Zookeeper service.

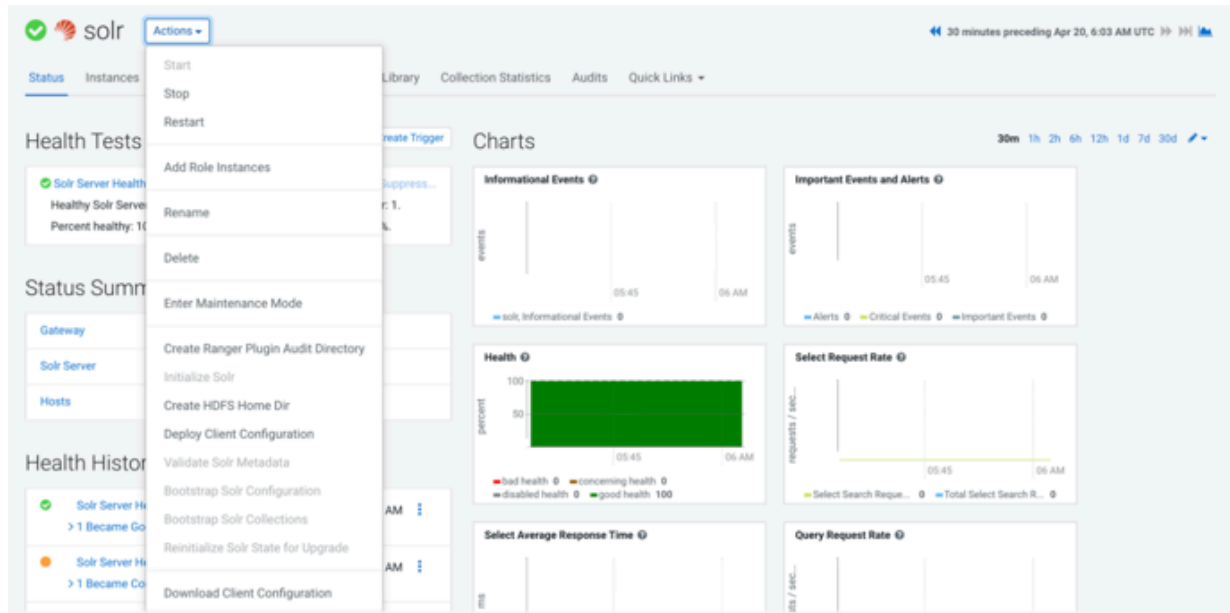
Procedure

1. Click solr in the Cloudera Manager UI.
2. Go to the Configuration tab.
3. Locate the HDFS Service (hdfs_service) property and select the Core Configuration option.



4. Click Save Changes.

5. Click Actions Initialize Solr .



6. Go to the initial Cloudera Manager UI by clicking the Cloudera Manager icon at the top-left corner of the screen.

7. Click zookeeper.

8. Click Actions Start in the Zookeeper cluster window.

The Start dialog box appears.

9. Confirm start operation by clicking Start again.

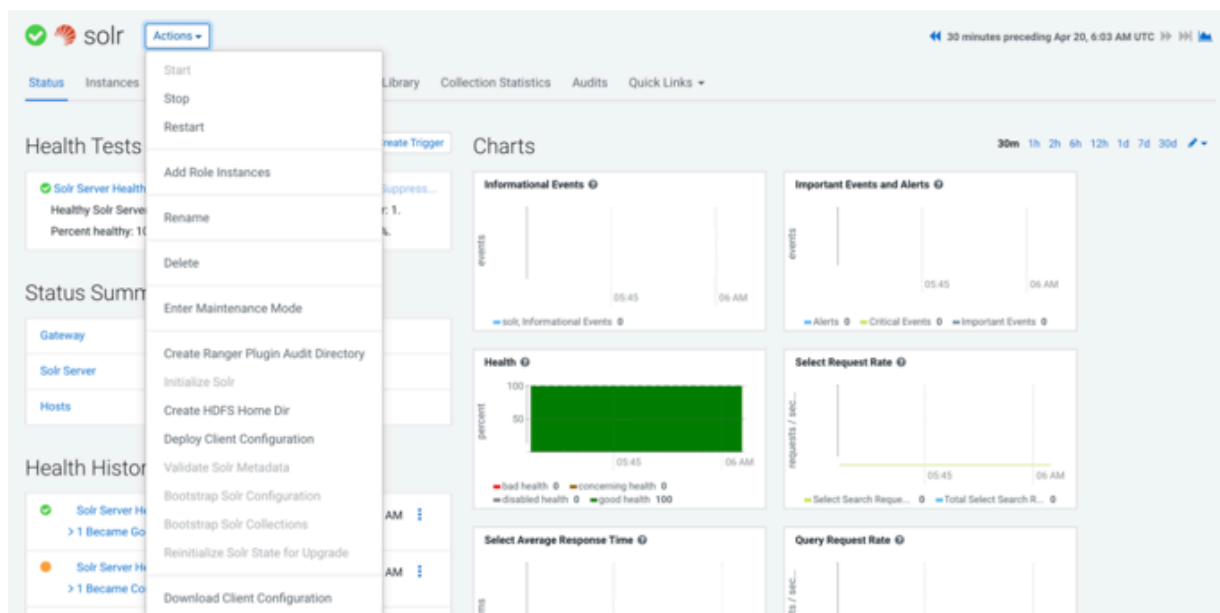
Initializing Solr

You need to execute Solr actions before you start using the Solr service to ensure that Solr is initialized correctly.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select Solr, and click Actions.
3. Execute the Initialize Solr action.

4. Execute the Create HDFS Home Dir action.



Configuring NiFi settings

You need to configure NiFi settings after you start NiFi successfully.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select NiFi.
3. Click NiFi Web UI.
4. Log in to NiFi.
5. On the NiFi UI, click Controller Settings.

The NiFi Settings screen appears.

6. Go to the Registry Clients tab.
7. Recheck or configure NiFi Registry URL on the NiFi UI to point to the correct hostname and port number.
8. Go to the Reporting Tasks tab.
9. Remove the AmbariReportingTask setting.

Configuring Kerberos for NiFi

After you enable Kerberos, as described in *Enable security*, you have to enable Kerberos for NiFi and configure the initial admin identity setting, if the initial admin identity setting was configured before migration.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select NiFi.
3. Go to the Configuration tab.
4. Search for the Enable Kerberos Authentication configuration value and enable it.
5. Optional. Search for the Initial Admin Identity configuration and specify the correct principal name.

Configuring Ranger for NiFi

If your cluster contains Ranger, you need to configure the Ranger service before starting NiFi. To create the default Ranger policies used by Cloudera Flow Management (CFM), you have to run the CreateMarkerFile command once.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select NiFi.
3. Go to the Configuration tab.
4. Search for the RANGER Service configuration and enable it.
5. Modify the `ranger.plugin.nifi.service.name` property to match the new Ranger service name.
6. Make sure that `ranger.plugin.nifi.policy.cache.dir` (Ranger NiFi Plugin Policy Cache Directory Path) is set to `/var/lib/nifi/policy-cache`.



Important: Do not set it to `/tmp`.

7. Confirm that NiFi is in a stopped state.

The `CreateMarkerFile` command can only be used when NiFi service is not running.

8. On the Cloudera Manager UI, go to NiFi service settings.
9. Run the Create Marker File action from the Actions dropdown.
After completing these actions and any other required NiFi-related post-configuration steps, starting the service automatically creates the default Ranger policies, if they are missing.

Modifying the service name in Ranger

If your NiFi service name in Ranger contains the `-` character, then you must change it to the `_` character.

Procedure

1. Go to Ranger Admin Web UI.
2. Go to resource based policies.
3. Click the edit button and modify the necessary characters in the service name and display name.
4. Click Save.

Migrating LDAP authentication configuration

If your NiFi used LDAP authentication in HDF cluster, you need to migrate the settings manually.

Procedure

1. Collect all necessary configuration for LDAP login provider.

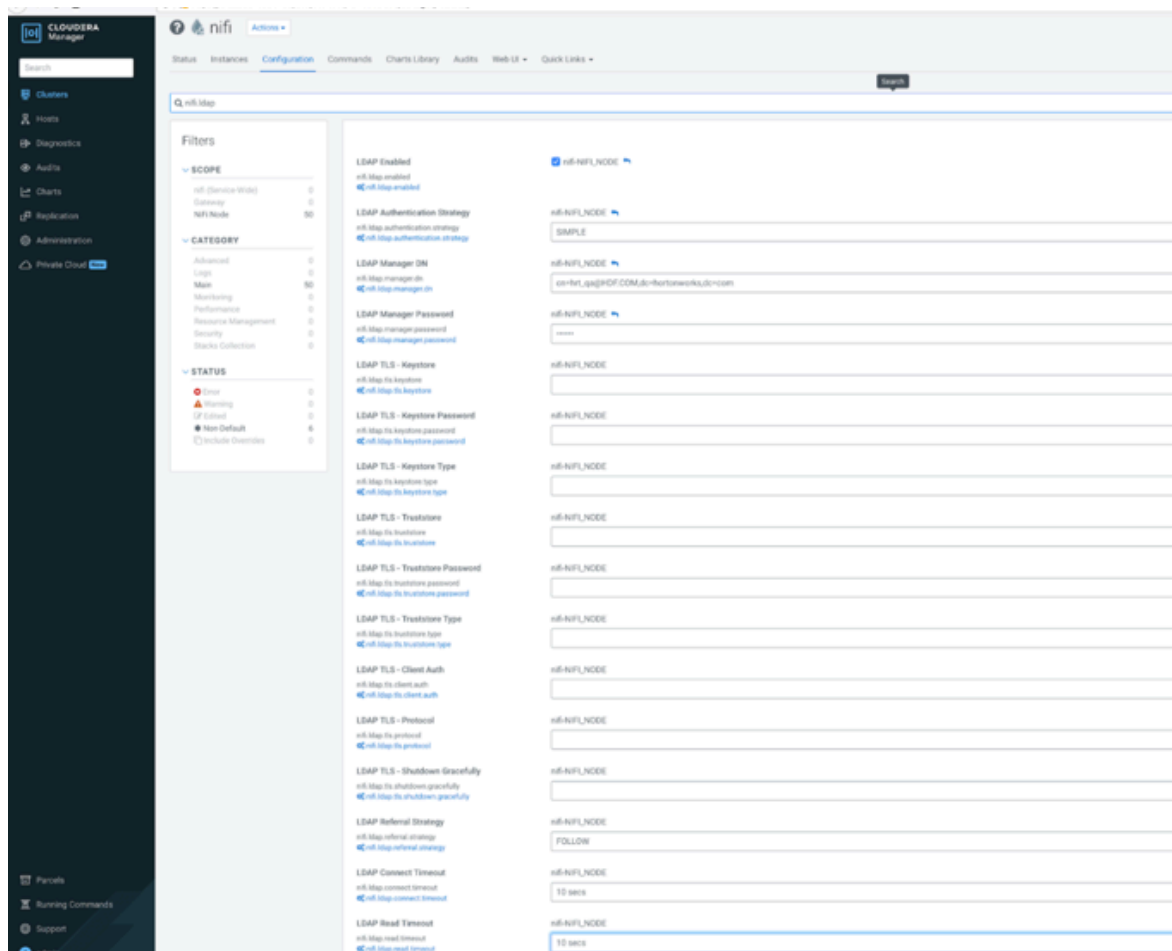
For that, you can check the old cluster configuration file or check the configuration in the Ambari UI:

```
cat /usr/hdf/current/nifi/conf/login-identity-providers.xml
```



Note: The passwords are encrypted in the XML file and cannot be fetched.

2. Configure the NiFi-LDAP properties in the Cloudera Manager UI:



3. Set the `nifi.security.user.login.identity.provider` configuration value to `ldap-provider`.
4. Set the `nifi.ldap.enabled` configuration value to `true`.
5. Configure the value of the `nifi.initial.admin.identity` property.
6. Remove the new cluster NiFi `users.xml` and `authorizations.xml` files for NiFi to generate these XML files with proper values.

The default path for these files is `/var/lib/nifi/users.xml` and `/var/lib/nifi/authorizations.xml`.

Migrating file-based user handling and policies

If you use NiFi built-in file-based policy and user handling, then you have to migrate the content of the `users.xml` and `authorizations.xml` files. This should be done after LDAP migration, if NiFi used LDAP.

Procedure

1. Copy the user or group entries from `/var/lib/nifi/conf/users.xml` and add the entries into `/var/lib/users.xml` for every NiFi instance.
2. Copy the policy entries from `/var/lib/nifi/conf/authorizations.xml` and add the entries into `/var/lib/authorizations.xml` for every NiFi instance.

If you have only a few user entries and policy configurations, it is quicker to re-apply them through the NiFi UI, instead of synchronizing the old and the new users and authorization XML files.

Configuring YARN settings

You need to configure YARN settings before you start using the YARN service.

Procedure

1. Go to Cloudera Manager Clusters .
2. Select Yarn.
3. Click Configuration.
4. Set value for the ApplicationMaster Maximum Attempts configuration property.

The value should be the same as the Maximum Number of Attempts for MapReduce Jobs configuration property value.

5. Remove the spark2_shuffle property from yarn.nodemanager.aux-services.

Migrating Kafka Ranger policies

After the upgrade of HDF to CFM on CDP, you must configure Kafka-centric clusters that use Ranger. You need to set Kafka Ranger policies. You need to ensure that the Ranger service name property has identical value for both Kafka and Ranger.

Before you begin

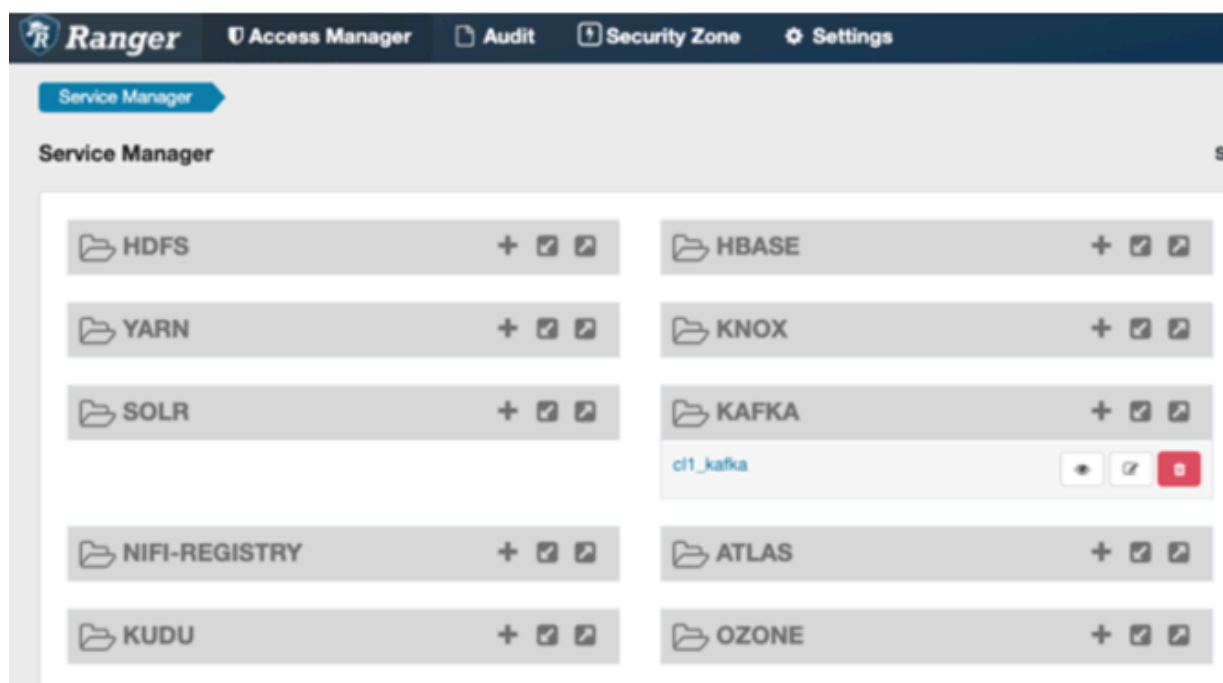
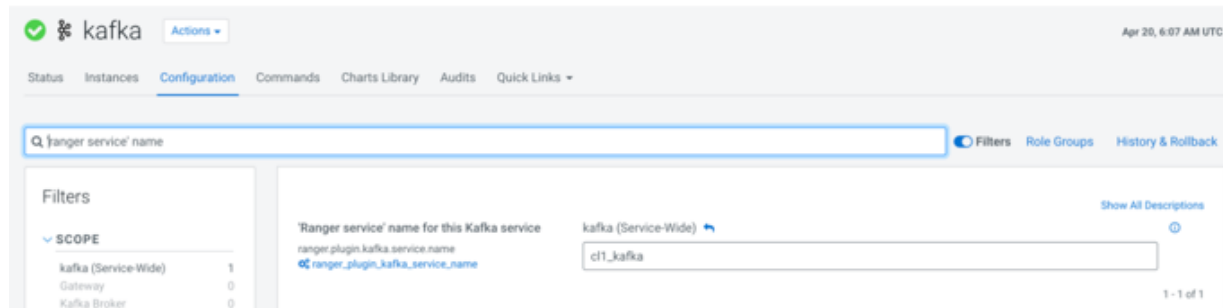
You have set core configuration service, and configured Ranger and SOLR settings.

Procedure

1. Click kafka in the Cloudera Manager UI.
2. Go to the Configuration tab.
3. Locate the 'Ranger service' name for this Kafka service property.

4. Ensure that the 'Ranger service' name for this Kafka service (ranger.plugins.kafka.service.name) kafka configuration matches the Ranger service name in the Ranger web UI.

For example, in the following images, the name of the Ranger service in both Kafka and Ranger is cl1_kafka.



To check the property name in Ranger:

- a. Click ranger in the Cloudera Manager UI.
- b. Click Ranger Admin Web UI.

The Ranger UI opens in another window.

- c. Click KAFKA service and check the Service Name property.
- d. If the service name is different, then set it to the same value as configured in the Kafka cluster.

Ranger

Access Manager

Audit

Security Zone

Settings

Service Manager

Edit Service

Edit Service

Service Details :

Service Name *

cl1_kafka

Display Name

cl1_kafka

Description

kafka repo

Active Status

☒ Enabled ☐ Disabled

Select Tag Service

Select Tag Service